IMS
Version 11

Commands, Volume 2: IMS Commands N - V

SC19-2431-05

IMS
Version 11

Commands, Volume 2: IMS Commands N - V

SC19-2431-05

Note
Before using this information and the product that it supports, be sure to read the general information under "Notices" on page 855.

This edition applies to IMS Version 11 (program number 5635-A02) and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 1974, 2015.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this information vii
Prerequisite knowledge vii
IMS function names used in this information . . . vii
How new and changed information is identified viii
How to read syntax diagrams viii
Accessibility features for IMS Version 11 x
How to send your comments x

Changes to the IMS library xi

Chapter 1. /NRESTART command 1

Chapter 2. /OPNDST command 11

Chapter 3. /PSTOP command 19

Chapter 4. /PURGE command 29

Chapter 5. QUERY commands 35
QUERY AREA command 36
QUERY DB command 49
QUERY DBDESC command 71
QUERY IMS command 79
QUERY IMSPLEX command. 84
QUERY LE command 93
QUERY LTERM command 100
QUERY MEMBER command 119
QUERY MSLINK command 129
QUERY MSNAME command 144
QUERY MSPLINK command 150
QUERY NODE command 156
QUERY ODBM commands 180
QUERY ODBM TYPE(ALIAS) command . . . 180
QUERY ODBM TYPE(CONFIG) command . . 183
QUERY ODBM TYPE(DATASTORE) command 187
QUERY ODBM TYPE(SCIMEMBER) command 191
QUERY ODBM TYPE(THREAD) command . . 194
QUERY ODBM TYPE(TRACE) command . . . 202
QUERY OLC command 204
QUERY OLREORG command 213
QUERY OTMADESC command 220
QUERY OTMATI command 226
QUERY PGM command 233
QUERY PGMDESC command 249
QUERY POOL command 258
QUERY RTC command 268
QUERY RTCDESC command 276
QUERY STRUCTURE command 283
QUERY TRAN command 287
QUERY TRANDESC command 318
QUERY USER command 336
QUERY USEREXIT command 354
QUERY USERID command 360

Chapter 6. QUEUE commands 371
QUEUE LTERM command 371
QUEUE TRAN command 376

Chapter 7. /QUIESCE command . . . 383

Chapter 8. /RCLSDST command . . . 385

Chapter 9. /RCOMPT command. . . . 387

Chapter 10. /RDISPLAY command . . 389

Chapter 11. /RECOVER commands 391
/RECOVER ADD command 391
/RECOVER REMOVE command 396
/RECOVER START command 400
/RECOVER STOP command 405
/RECOVER TERMINATE command 408

Chapter 12. /RELEASE command . . . 411

Chapter 13. /RESET command 413

Chapter 14. /RMxxxxxx commands 415

Chapter 15. /RSTART command . . . 427

Chapter 16. /RTAKEOVER command 433

Chapter 17. /SECURE command . . . 437

Chapter 18. /SET command 441

Chapter 19. /SIGN command 445

Chapter 20. /SMCOPY command . . . 451

Chapter 21. /SSR command 453

Chapter 22. /START commands. . . . 455
/START APPC command 456
/START AREA command 456
/START AUTOARCH command 460
/START CLASS command 461
/START DATAGRP command. 461
/START DB command 465
/START DC command 473
/START DESC command 474
/START ISOLOG command 474
/START LINE command 475
/START LTERM command 477

© Copyright IBM Corp. 1974, 2015

/START LUNAME command 478
/START MADSIOT command 480
/START MSNAME command 481
/START NODE command 481
/START OLDS command 483
/START OTMA command 484
/START PGM command 485
/START REGION command 487
/START RTC command 489
/START SB command 490
/START SERVGRP command 491
/START SLDSREAD command 492
/START SUBSYS command 493
/START SURV command 494
/START THREAD command 495
/START TMEM command 496
/START TRAN command 498
/START TRKARCH command 501
/START USER command 501
/START VGR command 503
/START WADS command 504
/START XRCTRACK command 505

Chapter 23. /STOP commands 507
/STOP ADS command 507
/STOP APPC command 508
/STOP AREA command 509
/STOP AUTOARCH command 513
/STOP BACKUP command 513
/STOP CLASS command 514
/STOP DATAGRP command 515
/STOP DB command 516
/STOP DC command 520
/STOP DESC command 521
/STOP LINE command 522
/STOP LTERM command 523
/STOP LUNAME command 525
/STOP MADSIOT command 526
/STOP MSNAME command 527
/STOP NODE command 528
/STOP OLDS command 530
/STOP OTMA command 531
/STOP PGM command 532
/STOP REGION command 533
/STOP RTC command 541
/STOP SB command 541
/STOP SERVGRP command 542
/STOP SLDSREAD command 543
/STOP SUBSYS command 544
/STOP SURV command 545
/STOP THREAD command 546
/STOP TMEM command 549
/STOP TRAN command 551
/STOP USER command 553
/STOP VGR command 555
/STOP WADS command 555
/STOP XRCTRACK command 556

Chapter 24. /SWITCH command . . . 557

Chapter 25. TERMINATE commands 563
TERMINATE OLC command 563
TERMINATE OLREORG command 575

Chapter 26. /TEST command 581

Chapter 27. /TRACE commands . . . 585
/TRACE EXIT command 586
/TRACE LINE command 587
/TRACE LINK command 589
/TRACE LUNAME command 592
/TRACE MONITOR command 593
/TRACE NODE command 596
/TRACE OSAMGTF command 600
/TRACE PI command 600
/TRACE PGM command 603
/TRACE PSB command 605
/TRACE TABLE command 606
/TRACE TCO command 610
/TRACE TIMEOUT command 611
/TRACE TMEMBER command 612
/TRACE TRAN command 614
/TRACE TRAP command 615
/TRACE UNITYPE command 616

Chapter 28. /UNLOCK commands . . . 619
/UNLOCK DB command 620
/UNLOCK LTERM command 621
/UNLOCK NODE command 622
/UNLOCK PGM command 623
/UNLOCK PTERM command 624
/UNLOCK SYSTEM command 625
/UNLOCK TRAN command 627

Chapter 29. UPDATE commands . . . 629
UPDATE AREA command 629
UPDATE DATAGRP command 643
UPDATE DB command 660
UPDATE DBDESC command 686
UPDATE IMS command 692
UPDATE LE command 701
UPDATE MSLINK command 707
UPDATE MSNAME command 717
UPDATE MSPLINK command 724
UPDATE ODBM commands 729
UPDATE ODBM START(CONNECTION) command 729
UPDATE ODBM START(TRACE) command . . 733
UPDATE ODBM STOP(CONNECTION) command 735
UPDATE ODBM STOP(TRACE) command . . 738
UPDATE ODBM TYPE(CONFIG) command . . 741
UPDATE OLREORG command 745
UPDATE OTMADESC command 750
UPDATE PGM command 755
UPDATE PGMDESC command 769
UPDATE POOL command 779
UPDATE RTC command 782
UPDATE RTCDESC command 789
UPDATE TRAN command 795

UPDATE TRANDESC command 829
Chapter 30. /VUNLOAD command . . 853
Notices 855
Privacy policy considerations 857
Bibliography 859
Index 861

About this information

These topics provide command syntax and usage information for the IMS™ type-1 and type-2 commands /NRESTART through /VUNLOAD. The topics also describe the IMS command language and how to send commands to IMS in different environments. Information about all non-type-1 and non-type-2 IMS commands is in *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands*.

This information is available in IBM® Knowledge Center at www.ibm.com/support/knowledgecenter.

Prerequisite knowledge

Before using this information, you should have knowledge of either IMS Database Manager (DB) or IMS Transaction Manager (TM). You should also understand basic z/OS® and IMS concepts, your installation's IMS system, and have general knowledge of the tasks involved in project planning.

Recommendation: Before using this information, you should be familiar with the following resources:

- *IMS Version 11 Operations and Automation*
- *z/OS JES2 Commands*
- *z/OS JES3 Commands*
- *z/OS MVS™ System Commands*

You can learn more about z/OS by visiting the "z/OS basic skills education" topics in IBM Knowledge Center.

You can gain an understanding of basic IMS concepts by reading *An Introduction to IMS*, an IBM Press publication. An excerpt from this publication is available in the IBM Knowledge Center.

IBM offers a wide variety of classroom and self-study courses to help you learn IMS. For a complete list of courses available, go to the IMS home page at ibm.com/ims and link to the Training and Certification page.

IMS function names used in this information

In this information, the term HALDB Online Reorganization refers to the integrated HALDB Online Reorganization function that is part of IMS Version 11, unless otherwise indicated.

IMS provides an integrated IMS Connect function, which offers a functional replacement for the IMS Connect tool (program number 5655-K52). In this information, the term IMS Connect refers to the integrated IMS Connect function that is part of IMS Version 11, unless otherwise indicated.

© Copyright IBM Corp. 1974, 2015

How new and changed information is identified

New and changed information in most IMS library PDF publications is denoted by a character (revision marker) in the left margin. The first edition (-00) of *Release Planning*, as well as the *Program Directory* and *Licensed Program Specifications*, do not include revision markers.

Revision markers follow these general conventions:

- Only technical changes are marked; style and grammatical changes are not marked.
- If part of an element, such as a paragraph, syntax diagram, list item, task step, or figure is changed, the entire element is marked with revision markers, even though only part of the element might have changed.
- If a topic is changed by more than 50%, the entire topic is marked with revision markers (so it might seem to be a new topic, even though it is not).

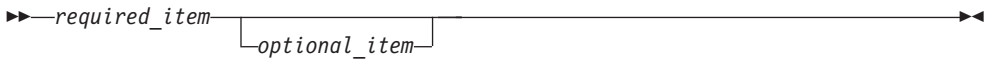
Revision markers do not necessarily indicate all the changes made to the information because deleted text and graphics cannot be marked with revision markers.

How to read syntax diagrams

The following rules apply to the syntax diagrams that are used in this information:

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following conventions are used:
  - The >>--- symbol indicates the beginning of a syntax diagram.
  - The ---> symbol indicates that the syntax diagram is continued on the next line.
  - The >--- symbol indicates that a syntax diagram is continued from the previous line.
  - The --->< symbol indicates the end of a syntax diagram.
- Required items appear on the horizontal line (the main path).

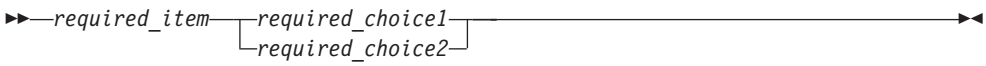
- Optional items appear below the main path.



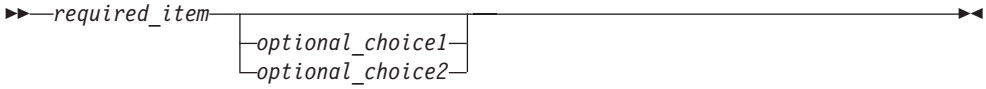
If an optional item appears above the main path, that item has no effect on the execution of the syntax element and is used only for readability.

- If you can choose from two or more items, they appear vertically, in a stack.

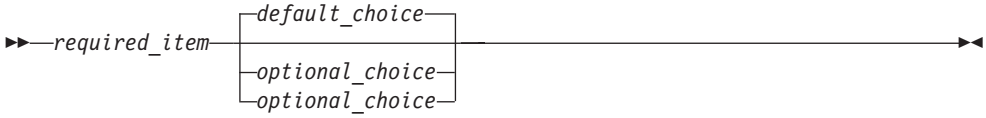
  If you *must* choose one of the items, one item of the stack appears on the main path.



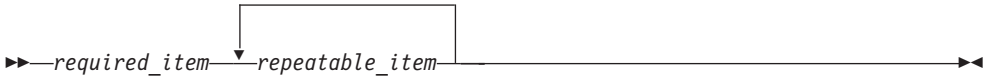
If choosing one of the items is optional, the entire stack appears below the main path.



If one of the items is the default, it appears above the main path, and the remaining choices are shown below.



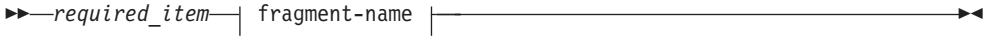
- An arrow returning to the left, above the main line, indicates an item that can be repeated.



If the repeat arrow contains a comma, you must separate repeated items with a comma.

A repeat arrow above a stack indicates that you can repeat the items in the stack.

- Sometimes a diagram must be split into fragments. The syntax fragment is shown separately from the main syntax diagram, but the contents of the fragment should be read as if they are on the main path of the diagram.



fragment-name:

- In IMS, a b symbol indicates one blank position.
- Keywords, and their minimum abbreviations if applicable, appear in uppercase. They must be spelled exactly as shown. Variables appear in all lowercase italic letters (for example, *column-name*). They represent user-supplied names or values.
- Separate keywords and parameters by at least one space if no intervening punctuation is shown in the diagram.

- Enter punctuation marks, parentheses, arithmetic operators, and other symbols, exactly as shown in the diagram.
- Footnotes are shown by a number in parentheses, for example (1).

Accessibility features for IMS Version 11

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility features

The following list includes the major accessibility features in z/OS products, including IMS Version 11. These features support:

- Keyboard-only operation.
- Interfaces that are commonly used by screen readers and screen magnifiers.
- Customization of display attributes such as color, contrast, and font size.

Keyboard navigation

You can access IMS Version 11 ISPF panel functions by using a keyboard or keyboard shortcut keys.

For information about navigating the IMS Version 11 ISPF panels using TSO/E or ISPF, refer to the *z/OS TSO/E Primer*, the *z/OS TSO/E User's Guide*, and the *z/OS ISPF User's Guide Volume 1*. These guides describe how to navigate each interface, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Related accessibility information

Online documentation for IMS Version 11 is available in IBM Knowledge Center.

IBM and accessibility

See the *IBM Human Ability and Accessibility Center* at www.ibm.com/able for more information about the commitment that IBM has to accessibility.

How to send your comments

Your feedback is important in helping us provide the most accurate and highest quality information. If you have any comments about this or any other IMS information, you can take one of the following actions:

- Click the **Feedback** link at the bottom of any IBM Knowledge Center topic.
- Send an email to imspubs@us.ibm.com. Be sure to include the book title and the publication number.

Changes to the IMS library

Changes to the IMS library for IMS Version 11 include minor title changes to all the publications, merging four sets of publications, a change to how *IMS Messages and Codes* are delivered, and a reorganization of *IMS Version 11 Exit Routines*. The library is no longer provided in BookManager format. In addition, as of the May, 2012, a master index is no longer provided.

The titles of all PDF publications in the IMS Version 11 library have been changed so that the publication titles match the contents view (navigation tree) titles in IBM Knowledge Center. For example, the *IMS Diagnosis Guide* has been renamed to *IMS Diagnosis*. In addition, the titles for the three *IMS Command Reference* PDF volumes have been renamed so that their titles better reflect the contents of each publication, as follows:

- *IMS Version 11 Commands, Volume 1: IMS Commands A-M*
- *IMS Version 11 Commands, Volume 2: IMS Commands N-V*
- *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands*

Four sets of publications have been streamlined:

- To eliminate redundancy and group relevant application programming planning information with associated application programming tasks, the *IMS Application Programming Planning Guide* has been merged with the *IMS Application Programming Guide*, in a new title: *IMS Version 11 Application Programming*.
- To group diagnosis reference information with associated diagnosis tasks, the *IMS Diagnosis Reference* has been merged with the *IMS Diagnosis Guide*, in a new title: *IMS Version 11 Diagnosis*.
- To group all system administration tasks, regardless of whether the tasks relate to using a single IMS or an IMSplex, the *IMS IMSplex Administration Guide* has been merged with the *IMS System Administration Guide*, in a new title: *IMS Version 11 System Administration*.
- To group system definition reference information with associated system definition tasks, the *IMS System Definition Reference* has been merged with the *IMS System Definition Guide*, in a new title: *IMS Version 11 System Definition*.

IMS Version 11 Exit Routines is now organized by the call interface and available services for each group of exit routines, as follows:

- IMS system exits, database manager exits, and transaction manager exit routines are grouped under “IMS control region exit routines.”These exit routines use IMS interfaces and callable services, as described in the topic “Overview of exit routines for IMS.”
- Exit routines that run in an address space with BPE support, including those for CQS, BPE-based DBRC, and most CSL components, are grouped under “Base Primitive Environment-based exit routines.” BPE-based exit routines use the interfaces and services provided by the Base Primitive Environment, as described in the topic “BPE user-supplied exit routine interfaces and services.”
- The CQS exit routines are now divided into two groups: those that run in the CQS address space and have access to BPE services are grouped under “BPE-based CQS user-supplied exit routines,” and those that execute in a CQS client address space are grouped under “CQS client exit routines.”

© Copyright IBM Corp. 1974, 2015

- Standard DBRC exit routines are grouped under “IMS system exit routines,” while the variants that run in BPE are group under “BPE-based DBRC exit routines.”
- CSL SCI exit routines are now divided into two categories: those that run in the CSL SCI address space are grouped under “BPE-based SCL SCI user exit routines,” and those that run in the address space of a registered IMSplex member are grouped under “CSL SCI IMSPlex member exit routines.”
- Exit routines that have specific guidelines, including those for IMS Connect, CQS clients, and CSL SCI IMSPlex members, are described in the sections for those routines.

Chapter 1. /NRESTART command

The /NRESTART command is used to cold start IMS or warm start IMS following an orderly termination accomplished with a /CHECKPOINT shutdown command.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 3
- "Usage notes" on page 6
- "Examples" on page 7

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

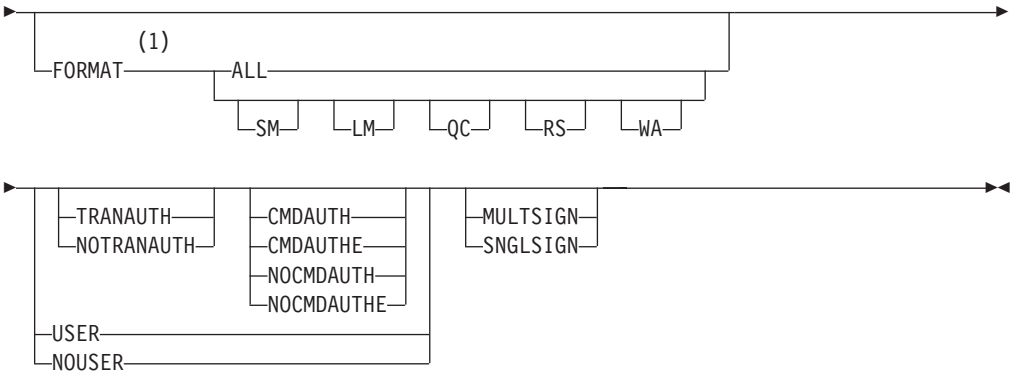
Table 1. Valid environments for the /NRESTART command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/NRESTART	X	X	X
BUILDQ	X		X
CHKPT	X	X	X
CMDAUTH	X		X
CMDAUTHE	X		X
FORMAT	X	X	X
MSDBLOAD	X		
MULTSIGN	X		X
NOBUILDQ	X		X
NOCMDAUTH	X		X
NOCMDAUTHE	X		X
NOTRANAUTH	X		X
NOUSER	X		X
SNGLSIGN	X		X
TRANAUTH	X		X
USER	X		X

Syntax

Cold start with no previous shutdown

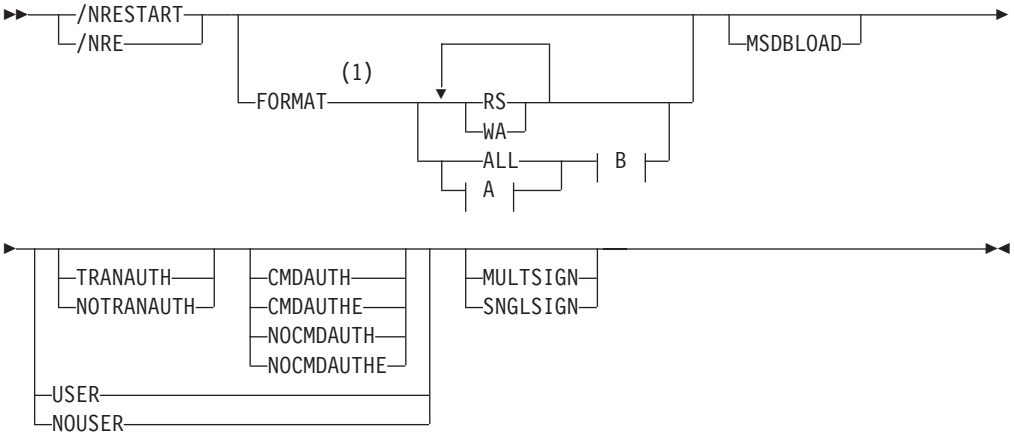
© Copyright IBM Corp. 1974, 2015



Notes:

1 The FORMAT keyword must be followed by at least one of the SM, LM, QC, RS, WA, or ALL parameters.

Warm start after a /CHECKPOINT FREEZE command



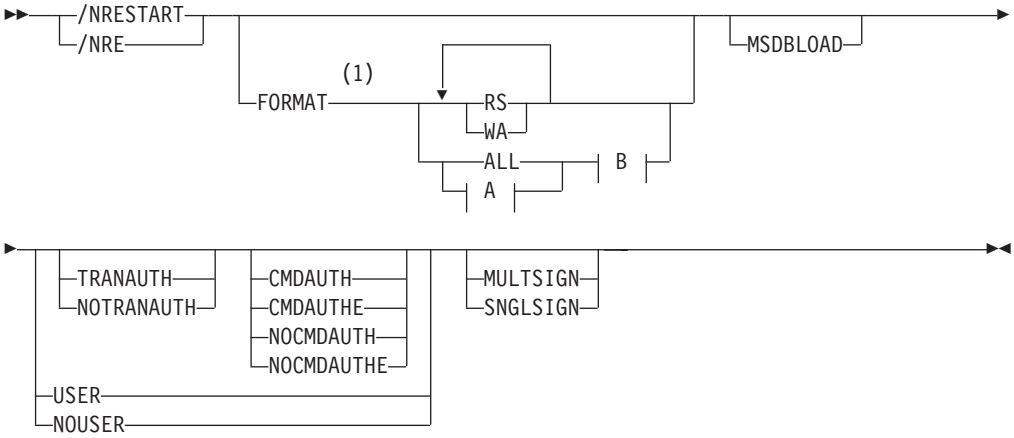
A:

B:

Notes:

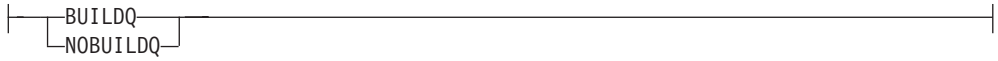
1 The FORMAT keyword must be followed by at least one of the SM, LM, QC, RS, WA, or ALL parameters.

Warm start after a /CHECKPOINT PURGE or /CHECKPOINT DUMPQ command



A:

B:



Notes:

1 The FORMAT keyword must be followed by at least one of the SM, LM, QC, RS, WA, or ALL parameters.

Keywords

The following keywords are valid for the /NRESTART command:

BUILDQ | NOBUILDQ

BUILDQ requests that the message queues dumped on the log be loaded into the message queue data sets. BUILDQ is optional for a warm start after a /CHECKPOINT PURGE or /CHECKPOINT DUMPQ.

The BUILDQ keyword must be included if the message queues are to be restored after being formatted. If the BUILDQ keyword is specified, the log from the last /CHECKPOINT DUMPQ or /CHECKPOINT PURGE is required, slowing down the restart process.

If /NRESTART FORMAT is specified without the BUILDQ keyword, the NOBUILDQ keyword must be specified. Specifying NOBUILDQ reformats the queues in question and all messages are lost.

If an /NRESTART BUILDQ command fails, and then the /ERESTART CHECKPOINT 0, /ERESTART COLDCOMM, or /ERESTART COLDSYS command is performed, the messages are lost. IBM IMS Queue Control Facility for z/OS can be used to recover the local message queues.

In a shared-queues environment, the BUILDQ keyword is ignored because the message queue data sets are not used.

CHECKPOINT
Identifies the shutdown/restart sequence. `CHECKPOINT 0` must be specified for a cold start.

CMDAUTH
Specifies that both signon (user identification verification) and command authorization for static and ETO terminals are in effect at the end of the emergency restart. (Command authorization is same as specifying RCF=S on the startup parameter.)

To specify CMDAUTH, you must specify one of the following:

- `TYPE=RACFTERM|RACFCOM|SIGNEXIT|TRANEXIT` on the SECURITY macro
- `RCF=A|Y|T|C|S` as an initialization EXEC parameter

CMDAUTHE
Specifies that command authorization for ETO terminals (same as RCF=S on the startup parameter) is in effect at the end of the emergency restart. CMDAUTHE also resets command authorization for static terminals, if it was set.

To specify CMDAUTHE, you must specify one of the following:

- `TYPE=RACFTERM|RACFCOM|SIGNEXIT|TRANEXIT` on the SECURITY macro
- `RCF=A|Y|T|C|S` as an initialization EXEC parameter

FORMAT
Specifies which queues or data sets should be formatted as part of the restart process when:

- A message queue or data set I/O error occurs.
- The size of a message queue or data set is to be changed.
- A message queue or data set is to be reallocated.

Specify one or more of the following or ALL:

SM Short-message queue

LM Long-message queue

QC Control record data set

RS Restart data set

WA Write-ahead data set

ALL All message queues (SM and LM) and data sets (QC, WA, and RS)

When FORMAT ALL is specified, do not also specify SM, LM, QC, WA, or RS. FORMAT ALL is only required at IMS initialization (first time use of the system).

You can specify any combination of SM, LM, QC, WA, and RS; for example, FORMAT LM RS.

In a shared-queues environment, the LM, SM, and QC parameters are ignored because the message queue data sets are not used. If you specify ALL, IMS does not attempt to format the message queue data sets.

The following table shows the environments in which the parameters are valid.

Table 2. /NRESTART FORMAT command parameter environments

Parameter	DB/DC	DBCTL	DCCTL
SM	X		X

Table 2. /NRESTART FORMAT command parameter environments (continued)

Parameter	DB/DC	DBCTL	DCCTL
LM	X		X
QC	X		X
RS	X	X	X
WA	X	X	X
ALL	X	X [1]	X

Note:

1. Supports only RS and WA parameters.

MSDBLOAD

Requests that the MSDBs be loaded from the z/OS sequential data set MSDBINIT instead of the MSDB checkpoint data set. Use the MSDBLOAD keyword only when an MSDB initial load is required; otherwise, omit it from the /NRESTART command. After you modify an MSDB DBD, you must specify MSDBLOAD on the next warm start of IMS in order for the changes to be effective.

MSDBLOAD is not required:

- For warm starts when the MSDB checkpoint data set is used.
- For a cold start because the MSDBs are loaded from the z/OS sequential data set MSDBINIT and the MSDB checkpoint data sets are formatted.

MULTSIGN

Permits multiple signons for each user ID.

In an IMSplex with Resource Manager and a resource structure, if MULTSIGN conflicts with the single user signon definition for the IMSplex, a warning message will be issued.

NOCMDAUTH

Resets command authorization on static and ETO terminals.

NOCMDAUTHE

Resets command authorization for static and ETO terminals. The command authorization is reset for static terminals because the command authorization for static terminals cannot exist without the command authorization for ETO terminals.

NOTRANAUTH

Turns off transaction authorization. NOTRANAUTH is not the opposite of TRANAUTH. TRANAUTH sets transaction authorization and also turns on signon (user identification verification).

If you specify NOTRANAUTH, it is rejected with an error message if either:

- SECLEVEL=FORCTRAN was specified on the system definition SECURITY macro.
- TRN=F was specified as a JCL EXEC parameter.

NOUSER

Specifies that none of the following is in effect at the end of the emergency restart:

- transaction authorization
- user identification verification
- command authorization

You can use /NRESTART NOUSER for a warm start, but if transaction or command authorization is set from the checkpoint data, NOUSER is ignored, and signon (user identification verification) is set on.

SNGLSIGN
Permits a single signon for each user ID.

In an IMSplex with Resource Manager and a resource structure, if SNGLSIGN conflicts with the single user signon definition for the IMSplex, a warning message will be issued.

TRANAUTH
Specifies both transaction authorization and user identification verification, with or without RACF®.

To specify TRANAUTH, you must specify one of the following:

- `TYPE=RACFTERM|RACFCOM|SIGNEXIT|TRANEXIT` on the SECURITY macro
- `RCF=A|Y|T|C|S` as an initialization EXEC parameter

USER
Specifies user identification verification. User identification verification means that signon is required by static terminals. This keyword has no effect on ETO terminals, because they are always required to sign on. User identification verification can be forced on by the TRANAUTH or CMDAUTH keyword

To specify USER, you must specify one of the following:

- `TYPE=RACFTERM|RACFCOM|SIGNEXIT|TRANEXIT` on the SECURITY macro
- `RCF=A|Y|T|C|S` as an initialization EXEC parameter

Usage notes

This command can be issued to an IMSplex using the Batch SPOC utility.

The /NRESTART command has three forms. The selected form depends on the following conditions:

- Whether a cold start or warm start is required
- In the case of warm starts, whether the previous orderly shutdown was accomplished with:
  - /CHECKPOINT FREEZE
  - /CHECKPOINT PURGE or DUMPQ

Attention: A cold start performed after a processing failure could cause processing against uncommitted data. To ensure data integrity, be sure necessary backout or recovery operations have been performed before restarting.

When IMS initializes, the system parameters used for this initialization can come from the IMS system generation, from a PROCLIB member, or from EXEC statements that can override both the defaults and the PROCLIB members. Therefore, message `DFS1929I` is displayed showing the system parameters used for this particular initialization. The system parameters are also written to the job log.

For an IMS cold start, the base security definition is created from the IMS system definition and EXEC parameter specifications. For an /NRESTART warm restart, the base security definition is created from the IMS checkpoint data.

To override the base security definitions on a cold start, the security keywords of the /NRESTART command must be used.

The SGN=, TRN=, and RCF= startup parameters can be overridden by the /NRESTART command using the security keywords shown in the following table. A brief description of the keywords is also included.

Table 3. Security keywords and their startup parameter equivalents

Keyword	Description	Startup parameter
CMDAUTH	RACF command authorization on static and ETO terminals only.	RCF=S
CMDAUTHE	RACF command authorization on ETO terminals only.	RCF=C
MULTSIGN	Permits multiple signons for each user ID.	SGN=M
NOCMDAUTH	Resets the command authorization on static and ETO terminals.	Not RCF=S
NOCMDAUTHE	Resets the command authorization on ETO terminals only.	Not RCF=C
NOTRANAUTH	Resets the transaction authorization.	Not TRN=F or Y
NOUSER	Resets user identification verification, transaction authorization, and command authorization.	Not SGN=F or Y (G or Z becomes M) Not TRN=F or Y Not RCF=C or S
SNGLSIGN	Permits a single signon for each user ID.	SGN=F and Y Not SGN=M (G or Z becomes F or Y)
TRANAUTH	Transaction authorization.	TRN=F or Y
USER	Sets user identification verification.	SGN=Y

Examples

The following are examples of the /NRESTART command:

Example 1 for /NRESTART command

This is an example of a cold start with new message queue data sets.

Entry ET:

```
/NRESTART CHECKPOINT 0 FORMAT ALL
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS994I *CHKPT 82274/114447**SIMPLE*
```

Explanation: IMS is started at 114447 (time) on 82274 (Julian date). A simple checkpoint is written on the system log. All message queue data sets are formatted. 82274/114447 is the checkpoint number.

Example 2 for /NRESTART command

This is an example of a warm start from a FREEZE checkpoint.

Entry ET:

```
/NRESTART
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82273/180000
DFS994I *CHKPT 82274/082217**SIMPLE*
```

Explanation: The restart is being performed from checkpoint 82273/180000, which was written at the most recent IMS shutdown. IMS is restarted at 082217 (time) on 82274 (Julian date). A simple checkpoint is written on this system log. 82274/082217 is the checkpoint number.

Example 3 for /NRESTART command

This is an example of a warm start to format WADS.

Entry ET:

```
/NRESTART FORMAT WA
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82119/230000
DFS994I *CHKPT 82120/101318**SIMPLE*
```

Explanation: The restart is being performed from checkpoint 82119/230000, which was written at the most recent IMS shutdown. IMS is restarted at 101318 (time) on 82120 (Julian date). A simple checkpoint is written on the system log. 82120/101318 is the checkpoint number.

Example 4 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint.

Entry ET:

```
/NRESTART BUILDQ
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82080/214240
DFS994I *CHKPT 82081/060000**SIMPLE*
```

Explanation: IMS is restarted at 060000 (time) on 82081 (Julian date) from checkpoint 82080/214240, which was written at the most recent IMS shutdown. 82081/060000 is the checkpoint number.

Example 5 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint. The large and small message queue data sets have been reallocated.

Entry ET:

```
/NRESTART BUILDQ FORMAT SM LM
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82170/085236
DFS994I *CHKPT 82170/085820**SIMPLE*
```

Explanation: IMS is restarted at 085820 (time) on 82170 (Julian date) from checkpoint 82170/085236, which was written at the most recent IMS shutdown. The large and small message queue data sets are reformatted. 82170/085820 is the checkpoint number.

Example 6 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint. An initial set of MSDBs is needed.

Entry ET:

```
/NRESTART BUILDQ MSDBLOAD
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82068/180000
DFS2554 MSDB MSDBHJ01 LOADED
DFS2554 MSDB MSDBHJ02 LOADED
DFS2554 MSDB MSDBAK01 LOADED
DFS2554 MSDB MSDBAK02 LOADED
DFS2554 MSDB MSDBPS01 LOADED
DFS994I *CHKPT 82069/080000**SIMPLE*
```

Explanation: IMS is restarted at 080000 (time) on 82069 (Julian date) from checkpoint 82068/180000, which was written at the most recent IMS shutdown. A simple checkpoint is written on the system log. 82069/080000 is the checkpoint number. An initial set of MSDBs is loaded from the z/OS sequential data set MSDBINIT.

Example 7 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint with a request for transaction command security.

Entry ET:

```
/NRESTART TRANCMDS
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82080/214240
DFS994I *CHKPT 82274/114447**SIMPLE*
```

Explanation: The master terminal operator is warm starting IMS and requesting that transaction command security be in effect at the completion of the normal restart.

Example 8 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint with a request to negate transaction command security.

Entry ET:

```
/NRESTART NOTRANCMDS
```

Response ET:

```
DFS2181I  CANNOT OVERRIDE FORCED COMMAND SECURITY
```

Explanation: The master terminal operator is warm starting IMS with a request to negate transaction command security. IMS system definition precludes the authority of the master terminal operator to enter the NOTRANCMDS keyword. IMS returns the DFS2181 message.

Example 9 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint. The data sets are formatted without rebuilding the message queues.

Entry ET:

```
/NRESTART FORMAT SM LM NOBUILDQ
```

Response ET:

```
DFS058 NRESTART COMMAND IN PROGRESS
```

Explanation: The master terminal operator is warm starting IMS with a request to reformat the data sets without rebuilding the message queues. IMS comes up, but all messages are lost.

Related reference:

IMS Queue Control Facility overview

Chapter 2. /OPNDST command

The /OPNDST command is a multisegment command that causes IMS to initiate a session with a VTAM® terminal.

For VTAM terminals, if the USER keyword is specified, the user is signed on automatically to the terminal after successful session initiation.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 12
- "Usage notes" on page 14
- "Examples" on page 15

Environment

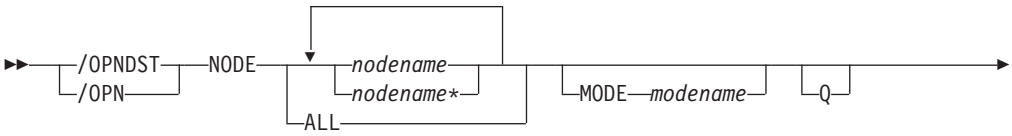
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 4. Valid environments for the /OPNDST command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/OPNDST	X		X
ID	X		X
LOGOND	X		X
MODE	X		X
NODE	X		X
Q	X		X
UDATA	X		X
USER	X		X
USERD	X		X

Syntax

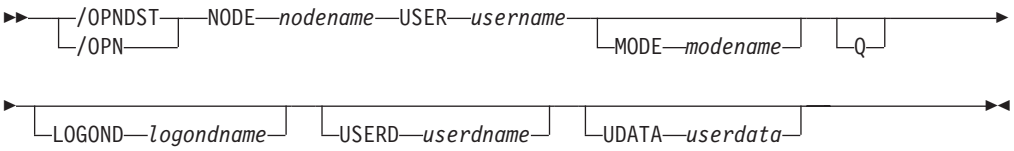
The /OPNDST command without the USER keyword: Use this form of the command for all static and ETO terminals except ETO SLU P and Finance terminals, ETO output-only devices, and all ISC parallel sessions.



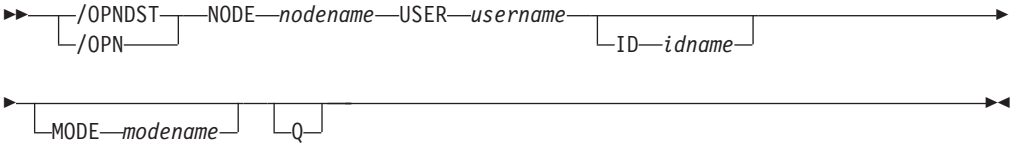
© Copyright IBM Corp. 1974, 2015

The /OPNDST command with the USER keyword for non-ISC: Use this form of the command for:

- ETO SLU P and Finance terminals
- ETO output-only devices, for example, 3284, 3286, and SLU P1 with a single component of PRINTER1



The /OPNDST command with the USER keyword for ISC VTAM: Use this form of the command for all ISC parallel sessions, both static and ETO.



Keywords

The following keywords are valid for the /OPNDST command:

ID Is applicable only if the USER keyword is specified and the node is a parallel session ISC VTAM node. ID must not be specified for a single session ISC node.

ID identifies the other system half-session qualifier. ID *idname* must be specified to open ISC nodes defined with users. The ID *idname* is passed to the other half-session with the session initiation request. If the other system is another IMS system, *idname* is the name of an ISC user in that system.

LOGOND
Indicates the logon descriptor used for session establishment. LOGOND is supported only for dynamic non-ISC nodes.

If a logon descriptor name is provided by the Logon exit routine (DFSLGNX0), the name provided by the exit routine overrides the name specified on the LOGOND keyword.

MODE
Identifies the LOGON MODE table entry that VTAM must use and can determine operating characteristics for certain VTAM terminals. If a list of node names is given, the MODE keyword is applied to each of them. The command checks whether each node has been defined to accept IMS initiated connections. If the node was defined as NOPNDST, the /OPNDST command accepts all the nodes except the node defined as NOPNDST.

NODE

Specifies the node with which IMS initiates a session.

The /OPNDST NODE ALL command opens sessions for all static terminals except ISC parallel sessions. The command has considerable concurrent activity, both for IMS and for VTAM. Ensure that the system has sufficient pool sizes, buffer sizes, and number of concurrent IMS tasks defined.

If the USER keyword is not specified in the command, the NODE parameter can be generic or ALL, or a range of static nodes, or there can be multiple NODE parameters. If a generic, ALL, or range of node names is specified, any nodes defined with users are ignored and flagged with an error message.

The /OPNDST NODE USER command logs on and signs on a user at the same time, except for ISC terminals. On ISC terminals, the session is allocated.

The following list includes sample /OPNDST NODE USER commands:

- To log on and sign on to a static non-ISC terminal:

  ```
  /OPNDST NODE nodename USER username UDATA userdata
  ```

  This command marks a statically defined terminal as signed on by the user.
- To log on and sign on to an ETO non-ISC terminal:

  ```
  /OPNDST NODE nodename USER username MODE modename
  LOGOND logondname USERD userdname UDATA userdata
  ```

  This command creates the terminal and user structures, and allocates the newly created user structure to the terminal structure created to indicate signed on status.
- To log on and sign on to a static ISC terminal:

  ```
  /OPNDST NODE nodename USER username ID idname
  ```

  This command finds the subpool structure and allocates the subpool (user) structure to the statically defined ISC terminal.
- To log on and sign on to an ETO ISC VTAM terminal:

  ```
  /OPNDST NODE nodename USER username ID idname MODE modename
  LOGOND logondname USERD userdname UDATA userdata
  ```

  This command creates the terminal and subpool (user) structure and allocates the newly created user structure to the terminal structure created. The command also signs on the user with the username (the username must be defined to RACF), and remains signed on until the user issues a /SIGN OFF command, or the session is terminated.

To restart failing ETO sessions (for example, ETO only or printer sessions), use the /OPNDST NODE USER command.

In an IMSplex, /OPNDST NODE specifies a VTAM node with which IMS initiates a session. Specify ROUTE(*imsid*), if you want to log the node onto a particular IMS. If ROUTE(*imsid*) is not specified, and /OPNDST is routed to all the IMS systems, IMS processes the command only on the IMS system designated as the command master. If the ROUTE keyword specifies multiple IMS systems so that the /OPNDST command is routed to more than one IMS systems, IMS processes the command only on the IMS system designated as the command master. On the other IMS systems, the /OPNDST command is rejected.

Q Causes IMS to request VTAM to queue SIMLOGON requests for VTAM/SNA-supported terminals.

The /OPNDST NODE Q command also allows IMS to request another subsystem to share a node (usually printers) with IMS. If the other subsystem is using the printer, VTAM queues the SIMLOGON request for IMS for the printer, schedules the owning subsystem's RELREQ VTAM exit, and acquires the printer for IMS after the current owning system releases the printer. Multiple requests for the same printer are queued by VTAM for processing.

The /DISPLAY NODE command is used to determine whether IMS has acquired the printer.

UDATA
Indicates the user data used with the signon.

The UDATA keyword is valid only if the USER keyword and parameter are also specified. The UDATA keyword is valid for static and dynamic users. It is not valid for ISC nodes. The user data can be up to 256 bytes long. Passwords can be mixed case or lowercase depending on what is specified on the PSWDC keyword in the DFSPBxxx IMS.PROCLIB member.

USER
Identifies the logical terminal user to be allocated to the half-session to be created for the ISC node that is specified on the NODE keyword.

For dynamic non-ISC users, it specifies the user ID to be signed on to the dynamic node *nodename*.

The USER keyword applies to ISC sessions when allocating a user to an ISC node, to dynamic users when signing a dynamic user on to a dynamic node, and to static user IDs when signing a user on to a static node.

USER username must be specified to open parallel session ISC nodes with users. It must not be specified for a single session ISC node.

Restrictions for using NODE and USER parameters together:

- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- The /OPNDST NODE USER commands are valid for ISC and non-ISC nodes and users.

USERD
Specifies the user descriptor to be used with the signon. It is valid only if the USER keyword and parameter are specified. USERD is only supported for dynamic users and is only valid for dynamic non-ISC nodes. The user descriptor can also be provided through the logon or signon exits.

Usage notes

All forms of logging-on a remote VTAM terminal to IMS, including the use of the /OPNDST command, do not work until the /START DC command has been entered and accepted by IMS.

All /OPNDST formats require an EOM indication to denote end-of-message. An EOS indication must be included for all segments that precede the last segment.

You can issue /OPNDST on the XRF alternate to restart a failed backup session for a class 1 ETO terminal. To do this, the node and the user structure must still exist

and be coupled together, and an active session must exist on the active system. If the node and the user are not coupled, or an active session does not exist on the active system, the command is rejected.

This command can be issued to an IMSplex using the Batch SPOC utility.

Examples

The following are examples of the /OPNDST command:

Example 1 for /OPNDST command

Entry ET:

```
/DIS NODE L3270*
```

Response ET:

```
NODE-USR TYPE   CID       RECD ENQCT DEQCT   QCT  SENT
L3270A   3277   00000000     0     0     0     0     0 IDLE C1INOP STATIC
L3270B   3277   00000000     0     0     0     0     0 IDLE C1INOP STATIC
L3270C   3277   08000002    44    45    45     0    80 CON STATIC
L3270D   3277   00000000     2     0     0     0     7 IDLE STATIC
*94307/145048*
```

Entry ET:

```
/OPNDST NODE L3270*
```

Response ET:

```
DFS058I  OPNDST COMMAND COMPLETED
```

Entry ET:

```
/DIS NODE L3270*
```

Response ET:

```
NODE-USR TYPE   CID       RECD ENQCT DEQCT   QCT  SENT
L3270A   3277   06000004     0     0     0     0     1 IDLE CON STATIC
L3270B   3277   04000005     0     0     0     0     1 IDLE CON STATIC
L3270C   3277   08000002    46    46    46     0    82 CON STATIC
L3270D   3277   04000006     2     0     0     0     8 IDLE CON STATIC
*94307/145750*

Explanation:  The nodes L3270A through L3270D are logged on to IMS.
```

Example 2 for /OPNDST command

Entry ET:

```
/OPNDST NODE DT327002 USER IMSUS01 MODE LU032NT4 USERD DFSUSER
UDATA= IMSPW01.
```

Response ET:

```
DFS058I 11:07:48 OPNDST COMMAND COMPLETED
```

Explanation: A session with dynamic node DT327002 is established by using mode table LU032NT4. User IMSUS01 is signed on to the node using user descriptor DFSUSER, username IMSUS01, and password IMSPW01.

Example 3 for /OPNDST command

Entry ET:

```
/OPNDST NODE DTSLU201 USER IMSUS01 MODE SLU2MOD1 USERD DFSUSER
LOGOND DFSSLU2 UDATA=IMSPW01.
```

Response ET:

```
DFS058I 11:07:48 OPNDST COMMAND COMPLETED
```

Explanation: DTSLU201 is logged on. A session with dynamic node DTSLU201 is established by using logon descriptor DFSSLU2 (type SLU2), and mode table SLU2MOD1. Dynamic user IMSUS01 is signed on to the node using user descriptor DFSUSER and password IMSPW01.

Example 4 for /OPNDST command

Entry ET:

```
/OPNDST NODE WEST-EAST
```

Response ET:

```
DFS058I  OPNDST COMMAND COMPLETED
```

Response RT:

```
DFS3650 TERMINAL CONNECTED TO IMS XXXXXXXX
```

Explanation: The nodes, WEST through EAST, are logged on to IMS.

Example 5 for /OPNDST command

The following set of commands illustrate the use of the MODE keyword on the /OPNDST command.

Entry ET:

```
/DIS NODE LUTYPEP1 MODE
```

Response ET:

```
NODE-USR TYPE   DEF MODETBL  ACT MODETBL
 LUTYPEP1 SLUP   DEFRESP
 *90179/100206*
```

Explanation: DEFRESP is the mode table name defined for node LUTYPEP1 at system definition. The session is not active so the ACT MODETBL field is blank.

Entry ET:

```
/OPN NODE LUTYPEP1.
```

Response ET:

```
DFS058I OPNDST COMMAND COMPLETED
```

Entry ET:

```
/DIS NODE LUTYPEP1 MODE
```

Response ET:

```
NODE-USR TYPE   DEF MODETBL  ACT MODETBL
 LUTYPEP1 SLUP   DEFRESP      DEFRESP
 *90179/100508*
```

Explanation: A mode table name was not specified with the /OPNDST command so the default value defined at system definition was used to initiate the session.

Entry ET:

```
/CLS NODE LUTYPEP1
```

Response ET:

```
DFS058I CLSDST COMMAND COMPLETED
```

Entry ET:

```
/DIS NODE LUTYPEP1 MODE
```

Response ET:

```
NODE-USR TYPE   DEF MODETBL  ACT MODETBL
 LUTYPEP1 SLUP   DEFRESP
 *90179/100630*
```

Explanation: Active mode table name displays as blank at normal session termination.

Entry ET:

```
/OPN NODE LUTYPEP1 MODE ALPHA.
```

Response ET:

```
DFS058I OPNDST COMMAND COMPLETED
```

Entry ET:

```
/DIS NODE LUTYPEP1 MODE
```

Response ET:

```
NODE-USR TYPE   DEF MODETBL  ACT MODETBL
 LUTYPEP1 SLUP  DEFRESP       ALPHA
 *90179/100805*
```

Explanation: The mode table name specified with the /OPNDST command (ALPHA) is used to initiate the session. The default value specified at system definition (DEFRESP) is overridden by the /OPNDST command.

Chapter 3. /PSTOP command

Use the /PSTOP command to stop the sending and receiving of messages to a particular communication line, terminal, or logical link.

You can also use this command to stop the scheduling of messages containing specific transaction codes, to allow the queuing of output messages and input messages to continue, and to perform validity checks of all parameters entered by the terminal operator.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 20
- "Usage notes" on page 23
- "Equivalent IMS type-2 commands" on page 23
- "Examples" on page 23

Environment

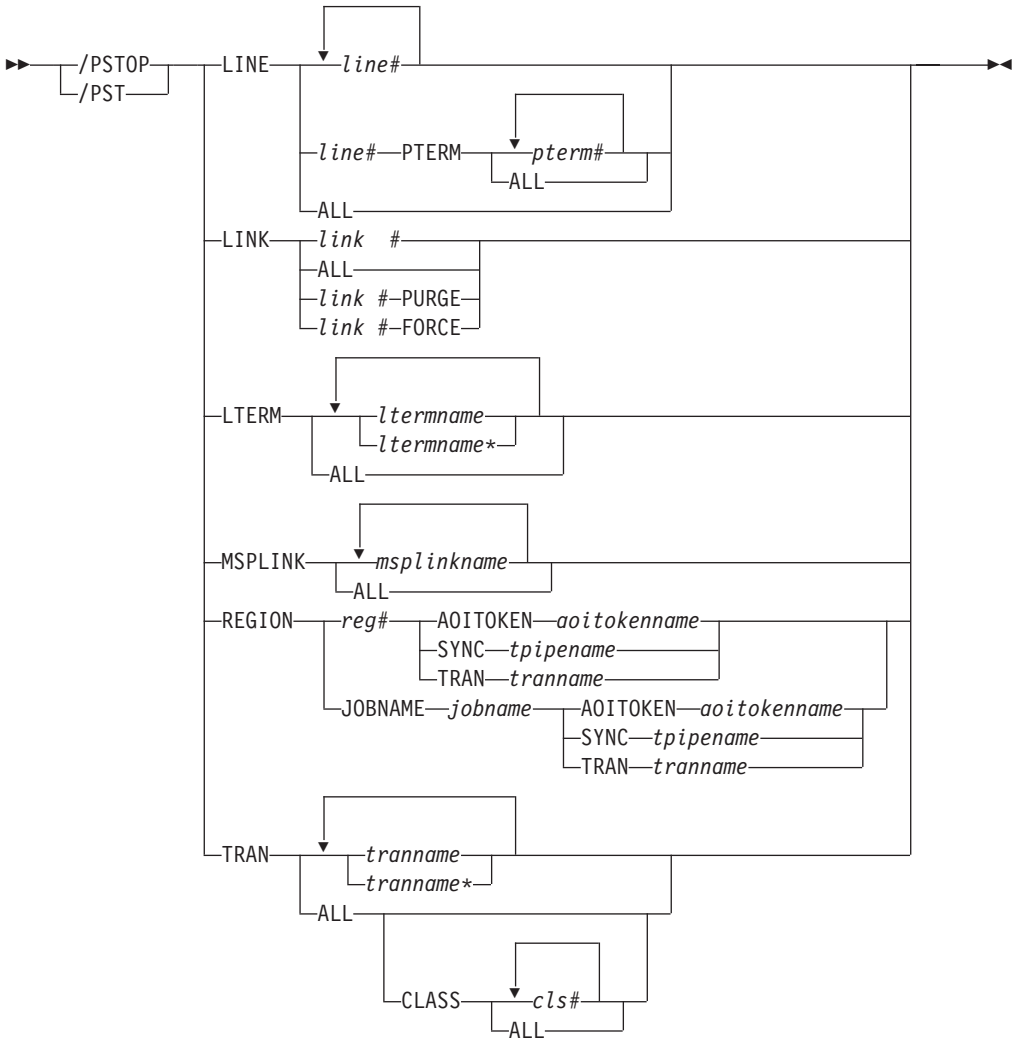
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 5. Valid environments for the /PSTOP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/PSTOP	X	X	X
AOITOKEN	X	X	X
CLASS	X		X
FORCE	X		X
JOBNAME	X	X	X
LINE	X		X
LINK	X		X
LTERM	X		X
MSPLINK	X		X
PTERM	X		X
PURGE	X		X
REGION	X	X	X
SYNC	X		X
TRAN	X		X

Syntax

© Copyright IBM Corp. 1974, 2015



Keywords

The following keywords are valid for the /PSTOP command:

LINE
Specifies the IMS communication line to stop.

LINK
Specifies the link to be stopped; the partner link in another IMS system stops itself and notifies the master terminal operator of that system..

PURGE
PURGE can be used only for one logical link whose physical link is channel-to-channel. PURGE must be used when the partner link is in a system that failed. Otherwise, the link will not become idle after it is stopped.

FORCE
The FORCE keyword is for VTAM links and is intended for use when an MSC VTAM link does not clean up and idle during normal PSTOP processing, even though VTAM terminated the session. It can be used, in

conjunction with some VTAM commands to idle and clean up the VTAM link within IMS, so that an /RSTART LINK can be issued to restart the link.

The operation of the command first determines if the link has begun PSTOP processing. (If so, the link will display as `PSTOPPED NOTIDLE`). Next it will test if the hang condition is due to an outstanding VTAM request that has not completed. If so, it will issue an inquire request to VTAM to determine if the session is inactive. In that case, IMS will simulate completion of the VTAM request and enable the link to complete PSTOP processing.

To determine if the session is still active to VTAM, issue a `DISPLAY NET,SESSIONS,LU1=applid1,LU2=applid2,SCOPE=ALL,LIST=ALL`, and note the SID of the session if it is active.

If the session is not active, and it has a PSTOPPED NOTIDLE status to IMS, then /PSTOP LINK x FORCE can be issued. If the session is still active to VTAM, then `VARY NET,TERM,SID=x,NOTIFY=YES,SCOPE=ALL,TYPE=FORCE` can be issued to VTAM to terminate the session.

Under normal conditions, the VTAM `VARY NET,TERM` command should terminate the session in VTAM and cause IMS to PSTOP and IDLE the link associated with the VTAM session. If the VTAM `VARY` command does terminate the VTAM session but does not PSTOP and IDLE the link, then the IMS /PSTOP LINK FORCE command can be used to complete the PSTOP and cleanup processing within IMS.

After PSTOP processing is completed on one side of the line, the other side should be displayed. If the other side is not in a PSTOP IDLE state, the operation should be repeated for the other IMS. When both sides are in the PSTOP IDLE state, the /RSTART LINK command can be issued.

IMS replies with `DFS058 /PSTOP LINK COMPLETED EXCEPT LINK x`, if the /PSTOP LINK x FORCE command cannot be executed because:

- The session is still active to VTAM.
- Normal PSTOP processing has not begun (issue /PSTOP without the FORCE keyword in this case).
- PSTOP processing is not completing because of some reason other than an incomplete VTAM request.

LTERM

Specifies the logical terminal that is to be stopped from sending and receiving messages.

The /PSTOP LTERM command has no effect on an LTERM that is in QLOCK state, or is a remote logical terminal. The LTERM parameter can be generic, where the generic parameter specifies existing LTERMs.

The /PSTOP LTERM command is valid only for LTERMs that belong to nodes that are logged on.

MSPLINK

Stops logons to an MSC VTAM physical link and enables the operator to issue the /MSASSIGN command to reassign logical links to the physical link. Any links in sessions that have not been stopped by the /PSTOP command are not affected by an /MSASSIGN command.

After the logical link assignments are complete, issue the /RSTART command to permit logons to the physical link.

The /PSTOP MSPLINK command does not apply to CTC or MTM links.

REGION

If the TRAN keyword is specified, the message region is not stopped. A QC status (no more messages) is returned to the application program currently active in the specified region. The scheduler continues to schedule available transactions in the referenced region.

The /PSTOP REGION command is ignored unless both of the following occur:

- An active transaction type is specified.
- The referenced message region is processing transactions with the wait-for-input option, or the region is an MPP.

If the AOITOKEN keyword is specified, the AO application in wait AOI token state is posted and receives AIB return code X'00000004' and reason code X'0000004C'.

If JOBNAME keyword is specified, the job name for the dependent region must be 1-8 alphanumeric or national ($,#,@) characters. The first character of the job name must be either alphabetic or national.

If the SYNC keyword is specified, the user application in wait synchronous callout state is posted and receives AIB return code X'00000100' with reason code X'0000010C'.

TRAN

Stops the scheduling of transactions; however, the transactions will continue to be processed until the limit count is reached. If the limit count is large, the processing interval will be long. The /DISPLAY command ascertains the status of the transaction; the /ASSIGN command alters the status of the transaction.

If a region is scheduled against a process stopped transaction and there are no more messages available for that transaction, the region does not wait for the next message (wait-for-input-mode). Instead, a QC status (no more messages) is returned to the application. If the region is scheduled and waiting for the next message when the command is entered, the region is notified and a QC status is returned to the application.

A batch message processing region (BMP) scheduled against wait-for-input (WFI) transactions returns a QC status code (no more messages) for /PSTOP REGION, /DBD, /DBR, or /STA commands only.

The /PSTOP command cannot stop the scheduling of Fast Path exclusive transactions but can be used to stop Fast Path potential transactions.

The /PSTOP TRAN command cannot be used for Fast Path exclusive transactions or CPI Communications driven transaction programs.

The TRAN parameter can be generic where the generic parameter specifies transactions that already exist.

In a shared-queues environment, the /PSTOP TRAN command will result in IMS deregistering interest for the transaction, which indicates that the transaction cannot be scheduled at that IMS.

Usage notes

If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and the /PSTOP command processes the rest of the parameters.

The /PSTOP command can be used to reset conditions previously established with the /START, /RSTART, /PURGE, or /MONITOR command.

In a single IMS system, or in the local system in a multiple system configuration, IMS system messages such as broadcast text and terminal status messages (`DFS059 TERMINAL STARTED`) are not affected by the /PSTOP command. In a multiple system configuration, broadcast messages are queued but not sent across stopped links.

This command can be issued to an IMSplex using the Batch SPOC utility.

Equivalent IMS type-2 commands

The following table shows variations of the /PSTOP command and the IMS type-2 commands that perform similar functions.

Table 6. Type-2 equivalents for the /PSTOP command

Task	/PSTOP command	Similar IMS type-2 command
Stops the scheduling of transactions.	/PSTOP TRAN *tranname*	UPDATE TRAN(*tranname*) START(Q) STOP(SCHD)
Stops logons to the physical link (only for MSC VTAM links).	/PSTOP MSPLINK *mplinkname* | ALL	UPDATE MSPLINK NAME(*msplinkname* | *) STOP(LOGON)

Examples

The following are examples of the /PSTOP command:

Example 1 for /PSTOP command

Entry ET:

```
/PSTOP LINE 4 PTERM 1
```

Response ET:

```
DFS058I  PSTOP COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL PSTOPPED
```

Explanation: LINE 4 PTERM 1 is not sent application program or message switch output and is not allowed to send input. Output messages for the terminal continue to be queued.

Example 2 for /PSTOP command

Entry ET:

```
/PSTOP LINE 4 6 200
```

Response ET:

```
DFS058I  PSTOP COMMAND COMPLETED EXCEPT LINE 200
```

Explanation: LINE 4 and LINE 6 are not allowed to send or receive messages. Message queuing continues. Line 200 is an invalid line number.

Example 3 for /PSTOP command

Entry ET:

```
/PSTOP LINK 2 3 4
```

Response ET:

```
DFS058I  PSTOP COMMAND COMPLETED
```

Response ET:

```
DFS2169I  DISCONNECTION COMPLETED ON LINK 2
```

Explanation: Logical link 2 is disconnected. This message is received for each logical link that is disconnected.

Response Remote MT:

```
DFS2161I  LINK 2 STOPPED BY PARTNER
DFS2161I  LINK 3 STOPPED BY PARTNER
DFS2161I  LINK 4 STOPPED BY PARTNER
```

Explanation: Logical links 2, 3, and 4 stop processing messages. Output queuing continues.

Response ET:

```
DFS2169I  DISCONNECTION COMPLETED ON LINK 3
```

Explanation: Logical link 3 is disconnected. This message is also received when logical link 4 disconnects.

Example 4 for /PSTOP command

Entry ET:

```
/PSTOP LINK ALL
```

Response ET:

```
DFS058I  PSTOP COMMAND COMPLETED
```

Response Remote MT:

```
A DFS2161 LINK n STOPPED BY PARTNER message
is received for each logical link that was
operational when /PSTOP was entered.
```

Explanation: Output to all logical links stops. Output queuing continues. Input is not allowed.

Response ET:

```
DFS2169I  DISCONNECTION COMPLETED ON LINK XXX
```

Explanation: As each logical link is disconnected, this message is received.

Example 5 for /PSTOP command

Entry ET:

```
/PSTOP LINK 1 2 3 PURGE
```

Response ET:

```
DFS2272I PURGE KEYWORD INVALID, ONLY ONE
CTC LINK ALLOWED
```

Explanation: Only one link can be specified with the PURGE keyword.

Example 6 for /PSTOP command

Entry ET:

```
/PSTOP LINK 2 PURGE
```

Response ET:

```
DFS2273I PURGE KEYWORD REJECTED, CURRENT STATUS
OF LINK IS NORMAL
```

Explanation: The partner system has not failed and the link appears to be working.

Example 7 for /PSTOP command

Entry ET:

```
/PSTOP LTERM APPLE, TREE
```

Response ET:

```
DFS058I  PSTOP COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL PSTOPPED
```

Explanation: The physical terminals associated with logical terminals APPLE and TREE are not sent output that is destined for logical terminals APPLE or TREE, or allowed to enter input. Output queuing continues.

Example 8 for /PSTOP command

Entry ET:

```
/PSTOP MSPLINK ALL
```

Response ET:

```
DFS058I  COMMAND COMPLETED
```

Explanation: All the VTAM physical links are stopped from receiving logons. Any links in session are not affected.

Example 9 for /PSTOP command

Entry ET:

```
/PSTOP TRAN SEED
```

Response ET:

```
DFS058I  PSTOP COMMAND COMPLETED
```

Explanation: Transaction code SEED can no longer be scheduled. Queuing of the transaction continues.

Example 10 for /PSTOP command

Entry ET:

```
/PSTOP TRAN ALL CLASS 3
```

Response ET:

```
DFS058I  PSTOP COMMAND COMPLETED
```

Explanation: All transactions associated with class 3 can no longer be scheduled. Queuing of the transactions continues.

Example 11 for /PSTOP command

Entry ET:

```
/PSTOP REGION 1 TRAN XYZ
```

Response ET:

```
DFS058I  PSTOP COMMAND IN PROGRESS
```

Response ET:

```
DFS0569I PSTOP COMPLETE FOR REGION 1 TRAN XYZ
DFS0566I PSTOP NOT VALID FOR TRAN XYZ
```

Explanation: If the DFS0569I message prints, processing of the transaction type, xyz, is stopped in message region 1. If the DFS0566I message prints, the command was ignored because the two required conditions were not satisfied.

Example 12 for /PSTOP command

Entry ET:

```
/PSTOP REGION 2 AOITOKEN AOITOK2
```

Response ET:

```
DFS058I PSTOP COMMAND IN PROGRESS
```

Response MT:

```
DFS0569I PSTOP OR STOP COMPLETE FOR REGION 2 AOIT AOITOK2.
```

Explanation: If the DFS0569I message prints, the AO application in region 2 waiting for a message for AOI token AOITOK2 is posted; the application receives AIB return code X'00000004' and reason code X'0000004C'.

Example 13 for /PSTOP command

Entry ET:

```
/PSTOP REGION 2 AOITOKEN  AOITOK2
```

Response ET:

```
 DFS058I PSTOP COMMAND IN PROGRESS
```

Response MTO:

```
DFS1190I REGION 2 NOT WAITING ON AOITOKEN AOITOK2
```

Explanation: If the DFS1190I message prints, the command is ignored because region 2 was not waiting on AOI token AOITOK2.

Example 14 for /PSTOP command

Entry ET:

```
/DIS ACTIVE REG
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS         CLASS
    1 MPP610C   TP    NQF1      PMVAPZ12 ACTIVE           1,  2,  3,  4
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
      DBTRGN    DBT   NONE
      DBRECTA9  DBRC
      DLIECTA9  DLS
      *96081/150611*
```

Explanation: Message processing program PMVAPZ12 is processing transaction NQF1. The job name of the region is MPP610C

Entry ET:

```
/PSTOP REGION JOBNAME MPP610C TRAN NQF1
```

Response ET:

```
DFS058I PSTOP COMMAND IN PROGRESS
DFS0569I PSTOP OR STOP COMPLETE FOR REGION 00001 TRAN NQF1
```

Response ET:

```
DFS058I PSTOP COMMAND IN PROGRESS
DFS0566I PSTOP NOT VALID FOR TRAN NQF1
```

Explanation: If the DFS0569I message is displayed, processing if the transaction type, NQF1, is stopped in message region 1. If the DFS0566I message is displayed, the command was ignored because the two required conditions were not satisfied.

Example 15 for /PSTOP command

Entry ET:

```
/DIS ACTIVE REG
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
    1 MPP1A     TP    APOL11    APOL1    WAIT-CALLOUT    1
                TMEM:   HWS1             TPIPE:  TPIPE1
      JMPRGN    JMP   NONE
      JBPRGN    JBP   NONE
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
      DBTRGN    DBT   NONE
      DBRZCSAJ  DBRC
      DLIZCSAJ  DLS
  *08235/173441*
```

Explanation: Message processing program APOL1 is processing transaction APOL11, which is waiting for a response to a synchronous callout request (WAIT-CALLOUT) from transaction pipe TPIPE1.

Entry ET:

```
/PSTOP REGION 1 SYNC TPIPE1
```

Response ET:

```
DFS058I PSTOP COMMAND IN PROGRESS
DFS0569I PSTOP OR STOP COMPLETE FOR REGION 00001 SYNC TPIPE1
```

Response ET:

```
DFS058I PSTOP COMMAND IN PROGRESS
DFS1190I REGION 00001 NOT WAITING ON SYNTPIPE TPIPE1
```

Explanation: If the DFS0569I message prints, the user application in wait synchronous callout state is posted; the application receives AIB return code X'00000100' and reason code X'0000010C'. If the DFS1190I message prints, the command is ignored because region 1 was not waiting for a response to a synchronous callout request from transaction pipe TPIPE1.

Related reference:

Chapter 15, "/RSTART command," on page 427

"UPDATE TRAN command" on page 795

Chapter 4. /PURGE command

The /PURGE command stops input for a particular communication line, terminal, or logical link path, or stops input messages destined for a particular transaction code.

Messages can be sent to the specified communication line or terminal, and transactions can still be scheduled.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 30
- "Usage notes" on page 31
- "Equivalent IMS type-2 commands" on page 31
- "Examples" on page 32

Environment

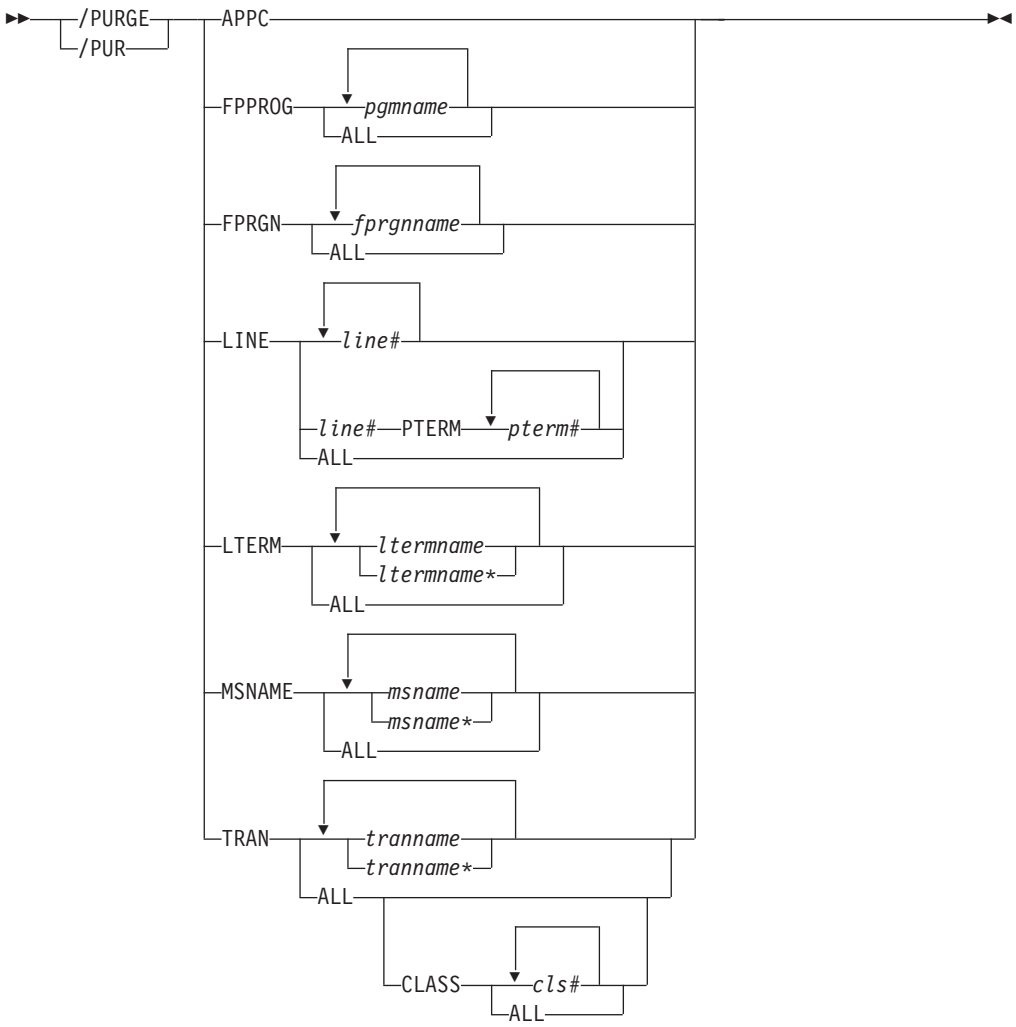
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 7. Valid environments for the /PURGE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/PURGE	X		X
APPC	X		X
CLASS	X		X
FPPROG	X		X
FPRGN	X		X
LINE	X		X
LTERM	X		X
MSNAME	X		X
PTERM	X		X
TRAN	X		X

Syntax

© Copyright IBM Corp. 1974, 2015



Keywords

The following keywords are valid for the /PURGE command:

APPC

Is used to purge incoming transactions. All new requests by APPC/z/OS to schedule a transaction in IMS are rejected with `TP_Not_Available_No_Retry`. Transactions that IMS has already received are processed normally. Sending of output to LU 6.2 devices proceeds normally. Because /PURGE APPC does not call to APPC/z/OS, the rejection of transaction scheduling is done by the schedule exit of IMS only.

Note: The sense code returned to the LU 6.2 remote device for an incoming ATTACH to a purged APPC/IMS system is determined by APPC/z/OS, and it might differ from release to release. In general, the remote LU 6.2 application should wait for a period of time after rejection before any attempts to reestablish a session with IMS.

The /PURGE APPC command sets the PURGING status and resets conditions previously set by the /START APPC command. The command is rejected if the APPC is already in DISABLED, FAILED, STOPPED, or CANCEL state.

FPPROG
Specifies the PSB name of the message-driven program to be terminated.

/PURGE takes message-driven programs out of wait-for-input mode and terminates them as soon as their load balancing group message queue is empty.

Use the FPPROG keyword with caution, because all Fast Path message-driven programs that are using a PSB with the same name will be terminated.

FPRGN
Specifies the region identifier of the message-driven program to be terminated.

LINE
Specifies the communication line for which input is to be stopped.

LTERM
Specifies the logical terminal for which input is to be stopped.

The /PURGE LTERM command is rejected for LTERMs in QLOCK state. (QLOCK indicates that the LTERM is locked from sending any further output or from receiving input that can create additional output for the same LTERM until the state is reset by a specific request received on the session.) /PURGE LTERM is also rejected for remote logical terminals. The LTERM supports generic parameters where the generic parameter specifies LTERMs that already exist.

The /PURGE LTERM command is valid only for LTERMs that belong to nodes that are logged on.

MSNAME
Specifies the logical link path in a multiple systems configuration for which input is to be stopped. The MSNAME keyword supports generic parameters.

TRAN
Specifies the transaction code for which input messages are to be stopped.

The TRAN parameter can be generic where the generic parameter specifies transactions that already exist.

Usage notes

The /PURGE command validity checks all parameters entered by the terminal operator. If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and the /PURGE command processes the rest of the parameters.

The /PURGE command can be used to reset conditions previously set by the /START, /RSTART, /STOP, /PSTOP, or /MONITOR command.

This command can be issued to an IMSplex using the Batch SPOC utility.

Equivalent IMS type-2 commands

The following table shows variations of the /PURGE command and the IMS type-2 commands that perform similar functions.

Table 8. Type-2 equivalents for the /PURGE command

Task	/PURGE command	Similar IMS type-2 command
Stops input messages for a particular transaction code.	/PURGE TRAN *tranname*	UPDATE TRAN NAME(*tranname*) START(SCHD) STOP(Q)

Examples

The following are examples of the /PURGE command:

Example 1 for /PURGE command

Entry ET:

```
/PURGE FPPROG ALL
```

Response ET:

```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: All message-driven programs are taken out of wait-for-input mode and terminated by PSB name as soon as their load balancing group message queue is empty.

Example 2 for /PURGE command

Entry ET:

```
/PURGE FPRGN ALL
```

Response ET:

```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: All message-driven programs are taken out of wait-for-input mode and terminated by region identifier as soon as their load balancing group message queue is empty.

Example 3 for /PURGE command

Entry ET:

```
/PURGE LINE 4
```

Response ET:

```
DFS058I PURGE COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL PURGING
```

Explanation: All physical terminals associated with line 4 can receive output sent to them but are not allowed to enter input.

Example 4 for /PURGE command

Entry ET:

```
/PURGE LINE 5 7 400
```

Response ET:

```
DFS058I  PURGE COMMAND COMPLETED EXCEPT LINE 400
```

Explanation: All physical terminals associated with line 5 and line 7 can receive output but are not allowed to enter input. Line 400 is an invalid line number.

Example 5 for /PURGE command

Entry ET:

```
/PURGE MSNAME BOSTON
```

Response ET:

```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: All messages from a terminal (primary requests), except messages continuing a conversation, will not be queued for the destinations represented by MSNAME BOSTON. This includes all messages destined for remote transactions with the SYSID of the MSNAME, and for remote logical terminals associated with this MSNAME.

Example 6 for /PURGE command

Entry ET:

```
/PURGE TRAN PIT, SEED
```

Response ET:

```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: Transactions PIT and SEED can still be scheduled but input for these transactions cannot be queued unless the input originates as output from an application program.

Example 7 for /PURGE command

Entry ET:

```
/PURGE TRAN ALL CLASS 2
```

Response ET:

```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: All transactions associated with class 2 are marked as purged. No further transactions are queued from terminals.

Related reference:

"UPDATE TRAN command" on page 795

"UPDATE MSNAME command" on page 717

Chapter 5. QUERY commands

Use the IMS QUERY commands to display information about IMS resources.

The QUERY commands return information based on the keyword specified. All of the QUERY commands are type-2 commands and can be issued from the OM API.

These commands can be issued through TSO SPOC or the Manage Resources options in the IMS Applications menu. These commands can also be issued to an IMSplex by using the Batch SPOC utility.

The value shown in the QCNT column of the command output has different meanings for different commands, as described in individual QUERY command topics.

QUERY commands are:

- "QUERY AREA command" on page 36
- "QUERY DB command" on page 49
- "QUERY DBDESC command" on page 71
- "QUERY IMS command" on page 79
- "QUERY IMSPLEX command" on page 84
- "QUERY LE command" on page 93
- "QUERY LTERM command" on page 100
- "QUERY MEMBER command" on page 119
- "QUERY MSLINK command" on page 129
- "QUERY MSNAME command" on page 144
- "QUERY MSPLINK command" on page 150
- "QUERY NODE command" on page 156
- "QUERY ODBM commands" on page 180
- "QUERY OLC command" on page 204
- "QUERY OLREORG command" on page 213
- "QUERY OTMADESC command" on page 220
- "QUERY OTMATI command" on page 226
- "QUERY PGM command" on page 233
- "QUERY PGMDESC command" on page 249
- "QUERY POOL command" on page 258
- "QUERY RTC command" on page 268
- "QUERY RTCDESC command" on page 276
- "QUERY STRUCTURE command" on page 283
- "QUERY TRAN command" on page 287
- "QUERY TRANDESC command" on page 318
- "QUERY USER command" on page 336
- "QUERY USEREXIT command" on page 354
- "QUERY USERID command" on page 360

© Copyright IBM Corp. 1974, 2015

QUERY AREA command

Use the QUERY AREA command, which is a type-2 command, to display information about DEDB areas and area data set information.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 37
- "Usage notes" on page 39
- "Equivalent IMS type-1 commands" on page 41
- "Output fields" on page 41
- "Return, reason, and completion codes" on page 43
- "Examples" on page 44

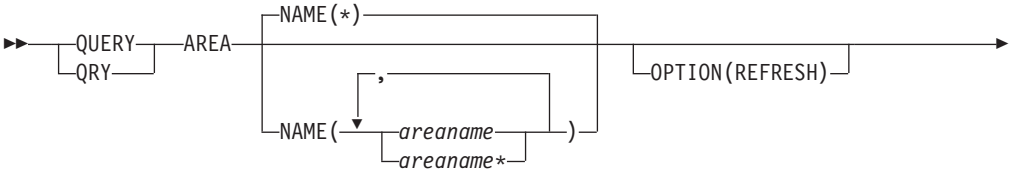
Environment

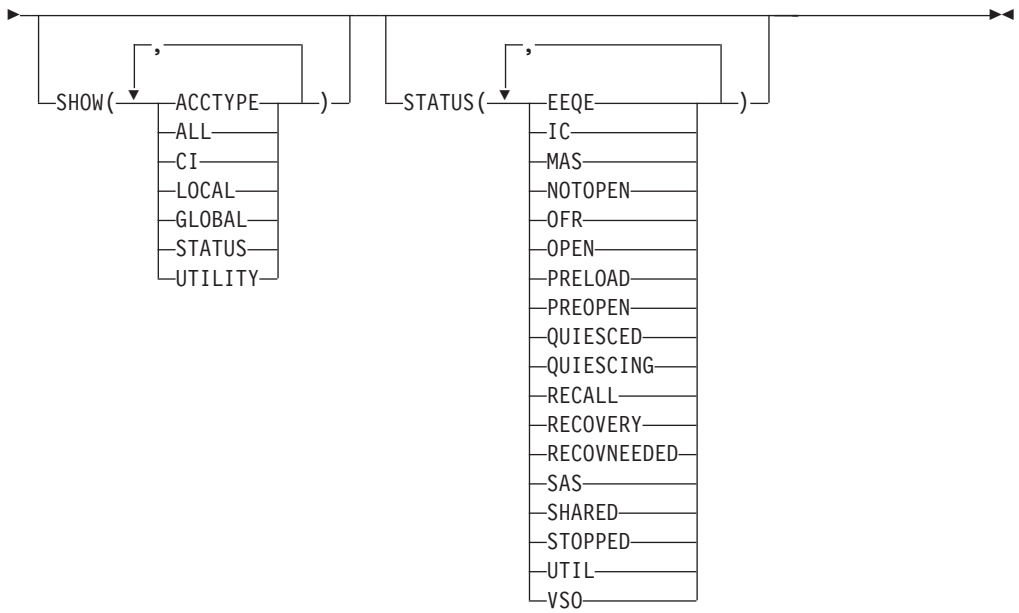
The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY AREA command and keywords can be issued.

Table 9. Valid environments for QUERY AREA command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUERY AREA	X	X	
NAME	X	X	
OPTION	X	X	
SHOW	X	X	
STATUS	X	X	

Syntax





Keywords

The following keywords are valid for the QUERY AREA command:

NAME()
Specifies the names of the specific areas that are to be processed or the group of areas whose names match the generic or wildcard parameter specified.

If the STATUS filter is not specified, all the area names that match the NAME parameter are returned. The NAME keyword is optional and the default is NAME(*).

OPTION(REFRESH)
Specifies the additional functions to be performed.

REFRESH
Refreshes the control interval information for the sequential dependent space and the independent overflow part of the direct addressable space for the area. OPTION(REFRESH) returns the CI information even if SHOW(CI) is not specified. OPTION(REFRESH) results in an I/O request that is performed only at the command master IMS if the area is open at the master. If the command is routed to multiple IMS systems in the IMSplex, all non-master IMS systems return local CI information. The age of the local information shown on the non-master IMS systems is at most as old as the value set on the IOVFI= IMS control region startup parameter.

The CI information is returned only if the area is open at the IMS. If the area is not open, blanks will be returned in the output CI columns SDAT, SDAU, LDAT, and LDAU. If the area is not open at the command master IMS, no refresh of the control intervals is performed. The QRY AREA NAME(areaname) SHOW(CI) OPTION(REFRESH) must be routed to the IMS where the area is open to get the current SDEP and IOVF CI information.

If the default NAME(*) is used with the REFRESH keyword, or if large numbers of areas are processed with the REFRESH keyword, performance may be affected depending on the size and number of areas involved. If large areas, numerous areas, or both are involved, the control regions processing the command may appear stopped.

SHOW()

Specifies the area output fields to be returned. The area name, the DEDB name, and the area data set information are always returned along with the name of the IMS that created the output for the area and the completion code.

The filters supported with the SHOW keyword are:

ACCTYPE

Returns DEDB area access when the area access has been changed explicitly by an UPDATE AREA START(ACCESS) SET(ACCTYPE()) command. If an area access is not changed explicitly by an UPDATE AREA command, the area access is blank. If none of the areas specified in the NAME() keyword of the QUERY AREA NAME() SHOW(ACCTYPE | ALL) command has an explicit area access value, then the area access (LAcc) output field column is not displayed.

Type of access to the DEDB area, which can be one of the following:

- BRWS - Read only
- EXCL - Exclusive
- READ - Read
- UPD - Update

ALL Returns all the output fields.

If global area status is maintained, the QUERY AREA command will return global information from the RM resource structure. The command master IMS returns the status on a separate response line. The area does not have to be defined at the command master.

CI Control intervals.

Returns the total and unused control intervals defined for the sequential dependent space and the total and unused control intervals for the independent overflow part of the direct addressable space.

LOCAL

For output fields that have both local and global values, this option returns only the local values. If used with another SHOW keyword to request a specific output field, this option requests that only the local value of the specified output field is returned. Local output is returned by each IMS that processes the command.

GLOBAL

For output fields that have both local and global values, this option returns only the global values. If used with another SHOW keyword to request a specific output field, this OPTION requests that only the global value of the specified output field is returned. IMS retrieves global information from RM. Global output is returned only by the command master IMS.

If global area status is maintained, the QUERY AREA command will return global information from the RM resource structure. The command master IMS returns the status on a separate response line. The area does not have to be defined at the command master.

STATUS
Local and global area status.

Global status is returned if global area status is maintained in RM. Global status is returned only by command master IMS and is returned on a separate response line.

If the area has an EEQE status, the count of I/O errors or write error EEQE for the area are also returned. If an area's status is OPEN, that status is not maintained in the RM resource structure.

If global area status is maintained, the QUERY AREA command will return global information from the RM resource structure. The command master IMS returns the status on a separate response line. The area does not have to be defined at the command master.

UTILITY
Returns the utility information about the utility that has the area open. The utility name, the total and available buffers in the private pool, and the utility UOW are returned.

STATUS()
Selects areas for display that match the NAME parameter and possess at least one of the specified area status locally.

The status filter allows for additional filtering by area status. The output returned when the STATUS filter is specified includes the status of the area that caused the area name to be displayed even if the SHOW(STATUS) is not specified.

Status parameters are the same as the values displayed in the QUERY AREA local status output column. See Table 11 on page 40.

Usage notes

The command can only be specified through the Operations Manager (OM) API and can only be processed by DB/DC and DBCTL environments. In addition, the QUERY AREA command is valid on the XRF alternate as well as the RSR tracker.

The command syntax for this command is defined in XML and is available to automation programs which communicate with OM.

How the SHOW keyword on QUERY AREA determines the type of output

The following table provides some examples of how the SHOW keyword determines the type of output returned on the QUERY AREA command.

Table 10. How the SHOW keyword on QUERY AREA determines the type of output

Form of SHOW keyword used	Type of output returned
SHOW(LOCAL)	Only those fields that are local to an IMS system. SHOW(ALL,LOCAL) provides the same output.
SHOW(GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM. SHOW(ALL,GLOBAL) provides the same output.

Table 10. How the SHOW keyword on QUERY AREA determines the type of output (continued)

Form of SHOW keyword used	Type of output returned
SHOW(ALL)	All of the output fields for those fields that have both local and global data. Both values are returned in the output.
SHOW(STATUS,GLOBAL)	Only global STATUS values.
SHOW(STATUS,LOCAL)	Only local STATUS values.
SHOW(STATUS)	Both local and global STATUS values.
SHOW(ALL,GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM. SHOW(GLOBAL) provides the same output.
SHOW(ALL,LOCAL)	Only those output fields that are local to an IMS system. SHOW(LOCAL) provides the same output.

QUERY AREA status

The following table lists the local and global area status conditions that may be returned when SHOW(STATUS) is specified.

Table 11. Status conditions for QUERY AREA

Status	Meaning
EEQE	Area has EEQEs.
IC	Area image copy is active.
MAS	Area is on a multi-area structure.
NONE	The area has no global status in the RM resource structure.
NOTOPEN	Area is not open.
OFR	Area has online forward recovery in progress to bring it up to current tracking level.
OPEN	Area is open. Not maintained in the RM resource structure.
PRELOAD	Area is defined to be preloaded.
PREOPEN	Area is defined to be preopened.
QUIESCED	The DEDB area named on the command is currently quiesced by a previous UPDATE DB START(QUIESCE) or UPDATE AREA START(QUIESCE) command.
QUIESCING	The DEDB area named on the command is currently undergoing quiesce by a previous UPDATE DB START(QUIESCE) or UPDATE AREA START(QUIESCE) command.
RECALL	Area is in recall.
RECOVERY	Area recovery in progress.
RECOVNEEDED	Area needs recovery.
SAS	Area is on a single area structure.
SHARED	Area is shared.
STA	The area is started globally.
STOACC	The area is stopped for access globally and is offline.
STOPPED	The area is stopped locally or globally.
UTIL	Area is open by a utility.
VSO	Area is a VSO area.

The following table lists the ADS status conditions that can be returned for an ADS associated with an AREA when SHOW(STATUS) is specified.

Table 12. ADS status conditions for QUERY AREA

Status	Meaning
COPY-PHASE	The CREATE utility is active on this ADS and is in the COPY phase. The CREATE utility must complete before any action can be processed for the ADS.
FORMAT-PHASE	The CREATE utility is active on this ADS and is in the FORMAT phase. The CREATE utility must complete before any action can be processed for the ADS.
LONGBUSY	Area in long busy state or long busy recovery mode.
PREOPEN-FAIL	XRF PREOPEN failed for this ADS.
SEVERE-ERROR	The ADS had a severe I/O error (write error to 2nd CI).
UNAVAIL	The ADS is marked unavailable because of I/O errors.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY AREA command and the type-1 IMS commands that perform similar functions.

Table 13. Type-1 equivalents for the QUERY AREA command

QUERY AREA command	Similar IMS type-1 command
QUERY AREA	/DIS AREA area1...arean | ALL, /DIS STATUS AREA

Output fields

The following table shows the QUERY AREA output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 14. Output fields for QUERY AREA command

Short label	Long label	Keyword	Scope	Meaning
ADS	ADSName	N/A	N/A	ADS name. The Area data set name or names associated with the AREA.
AREA	AreaName	N/A	N/A	Area name. The Area name is always returned.

Table 14. Output fields for QUERY AREA command (continued)

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned. See the return, reason, and completion codes table for QUERY AREA.
CCTXT	CCText	*error*	LCL	The completion code text that briefly explains the meaning of the completion code. The completion code text can be up to 32 bytes long.
DB	DBName	N/A	N/A	DEDB name. The DEDB name associated with the Area.
LDAT	Dir-T	CI	LCL	Local value of the total control intervals for the independent overflow part of the direct addressable space. This value only appears if the area is open.
LDAU	Dir-U	CI	LCL	Local value of the unused control intervals for the independent overflow part of the direct addressable space. This value only appears if the area is open and the IOVF count ITASK was not disabled when IOVFI=1 on the IMS Control Region startup procedure was specified. When the command is processed on an RSR tracking IMS system, no information will be returned. This value is refreshed during command processing if OPTION(IOVF) was entered on the QUERY AREA command. If OPTION(IOVF) was not specified, the value reflects the updated value from the last IOVF count ITASK.
LEQ	EQCnt	STATUS	LCL	Local value of the total control intervals for the independent overflow part of the direct addressable space. This value only appears if the area is open.
LPBA	PBuf-A	UTILITY	LCL	Available number of private buffers in private pool.
LPBT	PBuf-T	UTILITY	LCL	Total number of private buffers in private pool.
LSDT	SDep-T	CI	LCL	Local value of the total control intervals defined for the sequential dependent space. This value only appears if the area is open and SDEPs have been defined.
LSDU	SDep-U	CI	LCL	Local value of the unused control intervals defined for the sequential dependent space. This value only appears if the area is open, SDEPs have been defined, and unused SDEPs are available.
LSTT	LclStat	STATUS	LCL	Local area status. All area status conditions that apply are returned. Area status can be one or more of the status conditions described in Table 11 on page 40.
LUOW	UOW	UTILITY	LCL	The current utility UOW for HSREORG and HSSP, otherwise it is blank.
LUTIL	UtilName	UTILITY	LCL	Utility name that has area OPEN.

Table 14. Output fields for QUERY AREA command (continued)

Short label	Long label	Keyword	Scope	Meaning
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. IMS identifier of the IMS that built the output. The IMS identifier is always returned.
STT	Status	STATUS, GLOBAL	GBL	Global area status. Status can be: **STA** Area has a global status of started. **STO** Area has a global status of stopped. **STOACC** Area has a global status of stopped for access.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY AREA command. The OM return and reason codes that may be returned as a result of the QUERY AREA command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 15. Return and reason code for the QUERY AREA command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY AREA command completed successfully.
X'00000004'	X'00001010'	No resources were found to be returned. The resource name(s) specified may be invalid, or there were no resources that match the filter specified.
X'00000008'	X'0000200C'	The QUERY AREA command is not processed because no resources matched any status specified on the STATUS( ) keyword.
X'00000008'	X'00002014'	The QUERY AREA command is not processed because in invalid character is found in the area name parameter.
X'00000008'	X'00002040'	More than one filter or keyword value is specified on the QUERY AREA command. Either more than one keyword or an invalid combination of filters was specified. Check the input command and reenter the correct combinations.
X'0000000C'	X'00003000'	The QUERY AREA command was successful for at least one resource name. The QUERY AREA command was not successful for one or more resource names. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY AREA command are listed in Table 16 on page 44.

Table 15. Return and reason code for the QUERY AREA command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003004'	The QUERY AREA command was not successful for all the resource name(s) specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY AREA command are listed in Table 16.
X'00000010'	X'00004024'	The QUERY AREA command cannot be processed on a non-Fast Path system.
X'00000010'	X'00004025'	The QUERY AREA command is rejected because no Fast Path areas are defined.
X'00000014'	X'00005004'	The QUERY AREA command processing terminated as a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The QUERY AREA command processing terminated because of an internal error.

The following table includes an explanation of the completion codes. Errors unique to the processing of QUERY AREA command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 16. Completion codes for the QUERY AREA command

Completion code	Meaning
0	The QUERY AREA command completed successfully for the resource.
10	Resource not found. The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. Confirm that the correct spelling of the resource name is specified on the command.

Examples

The following are examples of the QUERY AREA command:

Example 1 for QUERY AREA command

In this example, the command returns the CI and STATUS information for the area, DB21AR0, from all the IMS systems in the IMSPlex. Any ADS information, if available, is also returned by each IMS.

TSO SPOC input:

```
QRY AREA NAME(DB21AR10) SHOW(STATUS,CI)
```

TSO SPOC output:

```
AreaName ADSName  MbrName  DBName     CC  SDep-T  SDep-U   Dir-T Dir-U EQCnt LclStat
DB21AR0           IMS2     DEDBJN21    0                                    PREOPEN,NOTOPEN
DB21AR0           SYS3     DEDBJN21    0    1303    1302   74    74          PREOPEN,OPEN,SHARED
DB21AR0  DB21AR01 SYS3                 0                                10
DB21AR0  DB21AR02 SYS3                 0                                10
```

OM API input:

```
CMD(QRY AREA NAME(DB21AR10) SHOW(STATUS,CI))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.132 16:10:52.861123</statime>
<stotime>2003.132 16:10:52.862301</stotime>
<staseq>B968A1B61BEC302F</staseq>
<stoseq>B968A1B61C35D38E</stoseq>
<rqsttkn1>USRT005 10091052</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>AREA            </kwd>
<input>QRY AREA NAME(DB21AR0) SHOW(CI,STATUS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="ADS" llbl="ADSName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="n" key="0" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="LSDT" llbl="SDep-T" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="INT" align="right" skipb="no" />
<hdr slbl="LSDU" llbl="SDep-U" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="INT" align="right" skipb="no" />
<hdr slbl="LDAT" llbl="Dir-T" scope="LCL" sort="n" key="0" scroll="yes"
 len="7" dtype="INT" align="right" skipb="no" />
<hdr slbl="LDAU" llbl="Dir-U" scope="LCL" sort="n" key="0" scroll="yes"
 len="7" dtype="INT" align="right" skipb="no" />
<hdr slbl="LEQ" llbl="EQCnt" scope="LCL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>AREA(DB21AR0 ) MBR(IMS2    ) DB(DEDBJN21) CC(   0) LSDT( ) LSDU( )
 LDAT( ) LDAU( ) LSTT(PREOPEN,NOTOPEN) </rsp>
<rsp>AREA(DB21AR0 ) MBR(SYS3    ) DB(DEDBJN21) CC(   0) LSDT(   1303)
 LSDU(   1302) LDAT(     74) LDAU(     74) LSTT(PREOPEN,OPEN,SHARED)
</rsp>
<rsp>AREA(DB21AR0 ) ADS(DB21AR01) MBR(SYS3    ) DB( ) CC(   0) LEQ(  10) </rsp>
<rsp>AREA(DB21AR0 ) ADS(DB21AR02) MBR(SYS3    ) DB( ) CC(   0) LEQ(  10) </rsp>
</cmdrspdata>
</imsout>
```

Example 2 for QUERY AREA command

In this example, the command returns all the areas that match the wildcard name and have a status of SHARED. The status is also returned. The ADS information for the AREAs is also returned if it is available. Command response lines are not returned from IMS2 because no AREAs match the status specified. A return and reason code is returned from IMS2.

TSO SPOC input:

```
QRY AREA NAME(DB21AR1*) STATUS(SHARED)
```

TSO SPOC output:

```
Log for . . : QRY AREA NAME(DB21AR1*) STATUS(SHARED)
IMSplex . . . . . : PLEX1
Routing . . . . . :
Start time. . . . : 2003.132 09:13:37.93
Stop time . . . . : 2003.132 09:13:37.94
Return code . . . : 0200000C
Reason code . . . : 00003000
Command master. . : IMS2
                Return     Reason
MbrName     Code       Code
--------    ------     ------
IMS2        00000008   0000200C
AreaName ADSName  MbrName  DBName     CC EQCnt LclStat
DB21AR10          SYS3     DEDBJN21    0       PREOPEN,OPEN,SHARED
DB21AR10 DB21AR10 SYS3                 0    10
DB21AR11          SYS3     DEDBJN21    0       PREOPEN,OPEN,SHARED
DB21AR11 DB21AR11 SYS3                 0    10
```

OM API input:

```
CMD(QRY AREA NAME(DB21AR1*) STATUS(SHARED))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.132 16:13:37.940282</statime>
<stotime>2003.132 16:13:37.940959</stotime>
<staseq>B968A2538A73A707</staseq>
<stoseq>B968A2538A9DF027</stoseq>
<rqsttkn1>USRT005 10091337</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS2    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000008</rc>
<rsn>0000200C</rsn>
<rsntext>No resources found</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMS2    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>AREA            </kwd>
<input>QRY AREA NAME(DB21AR1*) STATUS(SHARED) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="ADS" llbl="ADSName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="n" key="0" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
```

```
<hdr slbl="LEQ" llbl="EQCnt" scope="LCL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>AREA(DB21AR10) MBR(SYS3    ) DB(DEDBJN21) CC(   0)
 LSTT(PREOPEN,OPEN,SHARED) </rsp>
<rsp>AREA(DB21AR10) ADS(DB21AR10) MBR(SYS3    ) DB( ) CC(   0) LEQ( 10) </rsp>
<rsp>AREA(DB21AR11) MBR(SYS3    ) DB(DEDBJN21) CC(   0)
 LSTT(PREOPEN,OPEN,SHARED) </rsp>
<rsp>AREA(DB21AR11) ADS(DB21AR11) MBR(SYS3    ) DB( ) CC(   0) LEQ( 10) </rsp>
</cmdrspdata>
</imsout>
```

Example 3 for QUERY AREA command

In this example, the QUERY AREA SHOW(STATUS) command also returns the global status for the area. The global status is returned by the master IMS on a separate global response line.

TSO SPOC input:

```
QRY AREA NAME(DB21AR0,DB21AR1) SHOW(STATUS)
```

TSO SPOC output:

```
AreaName  ADSName  MbrName  DBName   CC  Status    LclStat
DB21AR0            IMS1     DEDBJN21  0  STOPPED
DB21AR0            IMS1     DEDBJN21  0            STOPPED,NOTOPEN
DB21AR0            IMS2     DEDBJN21  0            STOPPED,NOTOPEN
DB21AR1            IMS1     DEDBJN21  0  STA
DB21AR1            IMS1     DEDBJN21  0  OPEN
DB21AR1 DB21AR1    IMS1               0
DB21AR1            IMS2     DEDBJN21  0  OPEN
DB21AR1 DB21AR1    IMS2               0
```

Example 4 for QUERY AREA command

In this example, the QUERY AREA SHOW(STATUS) command also returns the global status for the area. The global status is returned by the master IMS on a separate global response line.

TSO SPOC input:

```
QRY AREA NAME(DB21AR0,DB21AR1) SHOW(STATUS)
```

TSO SPOC output:

```
AreaName  ADSName  MbrName  DBName   CC  Status    LclStat
DB21AR0            IMS1     DEDBJN21  0  STOPPED
DB21AR0            IMS1     DEDBJN21  0            STOPPED,NOTOPEN
DB21AR0            IMS2     DEDBJN21  0            STOPPED,NOTOPEN
DB21AR1            IMS1     DEDBJN21  0  STA
DB21AR1            IMS1     DEDBJN21  0            OPEN
DB21AR1 DB21AR1    IMS1               0
DB21AR1            IMS2     DEDBJN21  0            OPEN
DB21AR1 DB21AR1    IMS2               0
```

Example 5 for QUERY AREA command

The following example is of a query of the databases that have a status of quiesced.

TSO SPOC input:

```
QRY AREA NAME(*) SHOW(STATUS) STATUS(QUIESCED)
```

TSO SPOC output:

```
AreaName MbrName  DBName     CC LclStat
AXZY01   IM01     FPDBXYZ     0 OPEN,QUIESCED
AXZY01   IM02     FPDBXYZ     0 OPEN,QUIESCED
AXZY01   IM03     FPDBXYZ     0 OPEN,QUIESCED
```

Example 6 for QUERY AREA command

TSO SPOC input:

```
QUERY AREA NAME(DB21AR1*) SHOW(ACCTYPE)
```

TSO SPOC output:

```
AreaName ADSName  MbrName  DBName     CC  LAcc
DB21AR1           IMS1     DEDBJN21    0  READ
DB21AR1  DB21AR1  IMS1                 0
DB21AR1           IMS2     DEDBJN21    0  READ
DB21AR1  DB21AR1  IMS2                 0
DB21AR10          IMS1     DEDBJN21    0
DB21AR10          IMS2     DEDBJN21    0
DB21AR11          IMS1     DEDBJN21    0
DB21AR11          IMS2     DEDBJN21    0
```

Explanation: Database access for DEDB DEDBJN21 is UPD (Update). Area access for area DB21AR1 is READ. Area access for areas DB21AR10 and DB21AR11 are UPD, which they inherit from DEDB DEDBJN21 implicitly. Area access for areas DB21AR10 and DB21AR11 are not displayed because their area access is not changed by the UPDATE AREA START(ACCESS) SET(ACCTYPE()) command explicitly.

Example 7 for QUERY AREA command

TSO SPOC input:

```
QUERY AREA NAME(DB21AR2) SHOW(ACCTYPE)
```

TSO SPOC output:

```
AreaName ADSName  MbrName  DBName     CC
DB21AR2           IMS1     DEDBJN21    0
DB21AR2  DB21AR2  IMS1                 0
DB21AR2           IMS2     DEDBJN21    0
DB21AR2  DB21AR2  IMS2                 0
```

Explanation: Database access for DEDB DEDBJN21 is UPD (Update). Area access for area DB21AR2 is UPD, which is inherited from DEDB DEDBJN21 implicitly. Area access for area DB21AR2 is not displayed because its area access is not changed by the UPDATE AREA START(ACCESS) SET(ACCTYPE()) command explicitly. Because area DB21AR2 is the only area in the NAME() keyword and its area access has not been changed by the UPDATE AREA START(ACCESS) SET(ACCTYPE()) command, the output field column “LAcc” is not displayed.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY AREA command (Commands)

Command keywords and their synonyms (Commands)

QUERY DB command

Use the QUERY DB command, which is a type-2 command, to display information about databases.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 50
- "Usage notes" on page 54
- "Equivalent IMS type-1 commands" on page 55
- "Output fields" on page 55
- "Return, reason, and completion codes" on page 63
- "Examples" on page 64

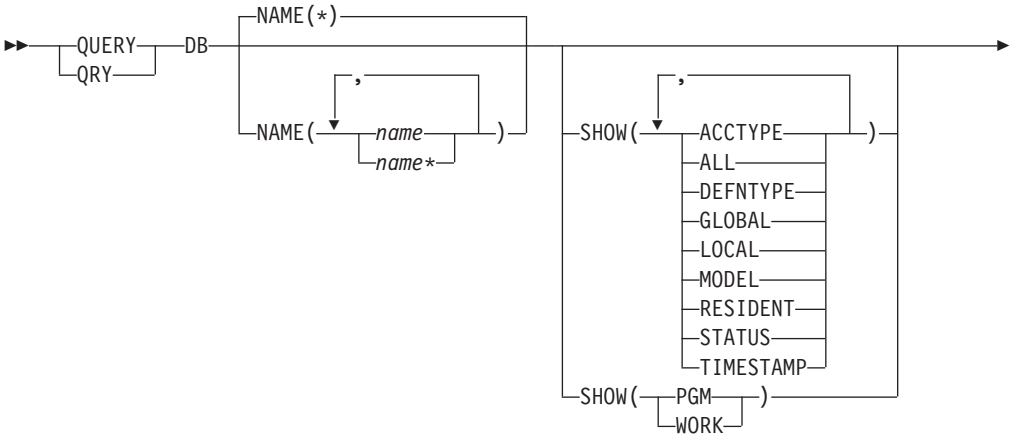
Environment

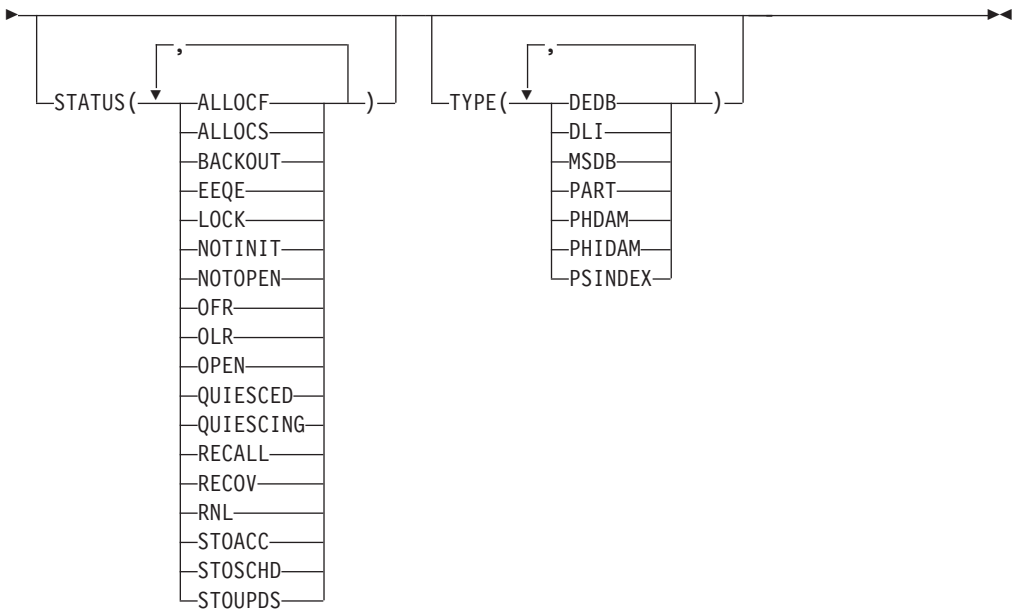
The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY DB command and keywords can be issued.

Table 17. Valid environments for QUERY DB command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUERY DB	X	X	
NAME	X	X	
SHOW	X	X	
STATUS	X	X	
TYPE	X	X	

Syntax





Keywords

The following keywords are valid for the QUERY DB command:

NAME()

Specifies the 1- to 8-character name of the database (DBD name). Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all database resources.

If the STATUS filter is not specified, all the database names that match the NAME parameter are returned.

The database names that match the generic or wildcard parameters are processed. Response lines are returned for all the databases names that are processed.

The database name specified can be a HALDB master or a HALDB partition. If the database name is the HALDB master, response lines are returned for the HALDB master and all of its partitions. If the database name is the HALDB partition, response lines are returned for the HALDB master and the partition name if the HALDB master has not been taken offline.

If the database name specified is a DEDB name, response lines are returned for the DEDB name and all the DEDB areas.

SHOW()

Specifies the database output fields to be returned. The database name and type are always returned, along with the name of the IMS that created the output for the database and the completion code. The filters supported with the SHOW keyword are:

ACCTYPE

Returns the type of access to the database or the area.

ALL
Returns all information about the database itself. Other SHOW keywords can be specified to return information about resources related to the database.

The command master IMS returns global information from the RM resource structure on a separate line, even if the database is not defined at the command master.

DEFNTYPE
Returns the definition type. This is how the resource was defined.

GLOBAL
For output fields that have both local and global values, this option returns only the global values. If used with another SHOW keyword to request a specific output field, this option requests that only the global value of the specified output field is returned. IMS retrieves global status information from RM. Global output is returned only by the command master IMS.

The command master IMS returns global status information from the RM resource structure on a separate line, even if the database is not defined at the command master.

If the database name is a HALDB master, global information is returned only for the master database, not for the HALDB partitions associated with the master. If the database name specified is a DEDB, global information is returned only for the DEDB, not the DEDB areas associated to the DEDB.

LOCAL
For output fields that have both local and global values, this option returns only the local values. If used with another SHOW keyword to request a specific output field, this option requests that only the local value of the specified output field is returned. Local output is returned by each IMS that processes the command.

MODEL
The model name and model type used to create this resource. The model name and model type are blanks for databases generated in the MODBLKS data set. The CREATE command specified without the LIKE keyword creates a resource using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSDB1 or user-defined. The CREATE command specified with the LIKE keyword creates a resource using a model. The resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

PGM
The names of the programs that reference the specified database.

The QRY DB SHOW(PGM) command will not show the area names if the DB is a DEDB or the partition names if the DB is a HALDB master.

The QRY DB SHOW(PGM) command will not show dynamic (DOPT) PSBs and database names for which the DOPT PSB has intent.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(PGM) individually.

RESIDENT
Returns the local runtime value for the resident option. The value returned is always RESIDENT(Y) for Fast Path DEDBs. The resident option definition is also shown, if it is different from the runtime value. The RESIDENT(Y) option takes effect at the next restart, unless the database was created or updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart.

STATUS
Returns the local and global database status.

If global status is maintained, the command master IMS returns global information from the RM resource structure on a separate line. The global status information is returned even if the database is not defined at the command master. For a description of the possible database status returned, see the STATUS keyword in the Output fields table under "Output fields" on page 55.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS) time, and import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

WORK
Work is in progress for the database that is specified on the NAME parameter and its associated resources. The QRY DB SHOW(WORK) command can be issued before a DELETE, IMPORT or UPDATE command to check for any work in progress for the specified database and any of its associated resources. Any work in progress might cause the subsequent DELETE, IMPORT or UPDATE command to fail. The QRY DB SHOW(WORK) command returns the status for the work in progress for the database that is specified on the NAME parameter.

Specifying SHOW(WORK) with NAME(*) might take a long time to process.

The QRY DB SHOW(WORK) command will not show the partition names if the database is a HALDB master.

Notes:

1. You cannot specify this filter with other SHOW filters; you must specify SHOW(WORK) individually.
2. The QRY DB SHOW(WORK) command is not valid on an XRF alternate.

STATUS()

Selects databases for display that match the NAME parameter and possess at least one of the specified database statuses. This selection allows for additional filtering by database status.

The output returned when the STATUS filter is specified includes the status of the database that caused the database name to be displayed even if SHOW(STATUS) is not specified.

ALLOCF
Sets the STATUS() filter to return information about databases that have allocation failure.

ALLOCS
Sets the STATUS() filter to return information about databases that are allocated successfully.

BACKOUT
Sets the STATUS() filter to return information about databases for which incomplete backout exists that prevents the use of the databases.

EEQE
Sets the STATUS() filter to return information about databases for which one or more extended error queue elements exist.

LOCK
Sets the STATUS() filter to return information about databases that are locked.

NOTINIT
Sets the STATUS() filter to return information about databases that are not initialized and therefore cannot be used.

NOTOPEN
Sets the STATUS() filter to return information about databases that are not open.

OFR
Sets the STATUS() filter to return information about databases that have online forward recovery in progress to bring them to the current tracking level.

OLR
The QUERY DB STATUS(OLR) command displays the status of all 'authorized' partitions with HALDB OLR in progress. The rate information is not returned. Instead, the rate information is returned on the QUERY OLREORG command output. A partition can be authorized by issuing a /START DB *partname* OPEN or UPDATE DB NAME(*partname*) OPTION(OPEN) command or by accessing the partition from an application. The SSYS record from a RECON listing can be used to determine whether a partition is authorized and to which subsystem.

OPEN
Sets the STATUS() filter to return information about databases that are open.

QUIESCED
Sets the STATUS() filter to return information about databases that are currently quiesced by a previous UPDATE DB START(QUIESCE) command.

QUIESCING
Sets the STATUS() filter to return information about databases that are undergoing quiesce by a previous UPDATE DB START(QUIESCE) command.

RECALL
Sets the STATUS() filter to return information about databases for which database recalls are in progress.

RECOV
Sets the STATUS() filter to return information about databases for which database recovery is in progress.

RNL
Sets the STATUS() filter to return information about DEDB databases for which randomizers are not loaded.

STOACC
Sets the STATUS() filter to return information about databases that are stopped for access locally or globally and are offline. The SHOW(STATUS) keyword on the QUERY DB command returns a status of STOACC for a DEDB. A status of STOACC indicates that the DEDB is stopped from further access because of a prior /DBR DB or UPDATE DB STOP(ACCESS) command.

STOSCHD
Sets the STATUS() filter to return information about databases that are stopped locally.

STOUPDS
Sets the STATUS() filter to return information about databases that are stopped for updates locally.

TYPE()
Selects databases for display that match the NAME parameter and specific TYPE filter. The supported TYPE filters are:

- DEDB
- DLI
- MSDB
- PART
- PHDAM
- PHIDAM
- PSINDEX

TYPE can be specified with the STATUS filter. If both TYPE and STATUS filters are specified, a response line is returned for each database that matches the NAME parameter and the TYPE and STATUS filters specified.

Usage notes

The command can be specified only through the OM API and can be processed only in DB/DC and DBCTL environments. In addition, QUERY DB is valid on the XRF alternate as well as the RSR tracker.

The command syntax for QUERY DB is defined in XML and is available to automation programs that communicate with OM.

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB except for the /START DB command and the UPDATE DB START(ACCESS) command. A HALDB partition is not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

The HALDB partition reflects conditions such as STOPPED (shown as STOACC or STOSCHD or STOUPDS on QUERY DB), LOCKED, or NOTOPEN. When a HALDB partition is stopped, it must be explicitly started again. The type-1 commands with the keyword ALL, type-2 commands with NAME(*), and commands against a HALDB master do not change the STOPPED (shown as STOACC, STOSCHD or STOUPDS on QUERY DB) and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is /DBR on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. On a /DISPLAY DB command, only the HALDB master displays STOPPED (each HALDB partition does not display STOPPED unless it was itself stopped). If a /DBR command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

Each partition inherits the access limitations of its HALDB master. If the /DBD command is issued against a HALDB master, all of its partitions close. A subsequent reference to any of the partitions results in the partition opening for input, although the partition's access might be UPDATE or EXCLUSIVE. The DBRC authorization state reflects the limited access.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY DB command and the type-1 IMS commands that perform similar functions.

Table 18. Type-1 equivalents for the QUERY DB command

QUERY DB command	Similar IMS type-1 commands
QUERY DB	/DIS DB dbname1...dbnamen | ALL, /DIS STATUS DB

Output fields

The following table shows the QUERY DB output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 19. Output fields for QUERY DB command

Short label	Long label	Keyword	Scope	Meaning
ACC	Acc	ACCTYPE, GLOBAL	GBL	Global access type information from RM, which can be one of the following values: **BRWS** Database has global access type of BRWS. **EXCL** Database has global access type of EXCL. **READ** Database has global access type of READ. **UPD** Database has global access type of UPD.
AREA	AreaName	N/A	N/A	Area name. The area name is returned if there are one or more response lines for DEDB areas in the output.
CC	CC	N/A	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned. Refer to the return, reason, and completion codes for QUERY DB.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DB	DBName	DB	N/A	Database name.
DFNT	DefnType	DEFNTYPE	N/A	Definition type, which can be one of the following values: **CREATE** Defined by a CREATE command. **IMPORT** Defined by an IMPORT command. **IMS** Defined by IMS. **MODBLKS** Defined by system definition in the MODBLKS data set. **UPDATE** Defined by system definition in the MODBLKS data set, but changed into a dynamic resource by an UPDATE command.

Table 19. Output fields for QUERY DB command (continued)

Short label	Long label	Keyword	Scope	Meaning
LACC	LAcc	ACCTYPE	LCL	Type of access to database or area, which can be one of the following. **BRWS** The database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data since a sharing IMS subsystem could be updating the database. The database is opened for read-only processing. **EXCL** The named database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC. **READ** The database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. Access type READ differs from access type BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing. **UPD** The database is available for update as well as read processing in the IMS subsystem.
LRSDNT	LRsdnt	RESIDENT	LCL	Local runtime value of the resident option. Indicates whether the database DMB resides in local storage. This value is not returned for database descriptors. **N** The DMB associated with the named database resource is not made resident in storage. If a database is defined as resident but encounters an error during IMS restart, N is returned. The DMB is loaded at scheduling time. **Y** The DMB associated with the named database resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the DMB and initializes it. A resident database is accessed from local storage, which eliminates I/O to the ACBLIB. In an online environment, the DMB control blocks are stored in the ACBLIB. If the DLI/SAS address space exists, DLI/SAS loads the DMB, otherwise, it is the IMS control region that loads it. This makes the DMB dependent on the existence of the corresponding database resource.
LSTT	LclStat	STATUS	LCL	Local database status. All database status conditions that apply are returned. Database status can be one or more of the status conditions described in Table 20 on page 60.
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. The IMS identifier of the IMS that built the output. The IMS identifier is always returned.

Table 19. Output fields for QUERY DB command (continued)

Short label	Long label	Keyword	Scope	Meaning
MDLN	ModelName	MODEL	N/A	Model name. Name of the resource or descriptor that is used as a model to create this resource. DFSDSDB1 is the IMS descriptor name for databases.
MDLT	ModelType	MODEL	N/A	Model type, either RSC or DESC. RSC means that the resource was created by using another resource as a model. DESC means that the resource was created by using a descriptor as a model.
PART	PartName	N/A	N/A	HALDB partition name. The partition name is returned if there is one or more response lines for HALDB partitions in the output.
PGM	PgmName	PGM	N/A	Program name that references the database.
RSDNT	Rsdnt	RESIDENT	LCL	Resident option. Indicates whether the database DMB is to reside in local storage at the next IMS restart. This value is only returned if the resident option definition is different from the resident option local runtime value.
STT	Status	STATUS, GLOBAL	GBL	Global status information from RM, which can be one of the following values: **ALLOC** Database has a global status of allocated. **LOCKED** Database is a global status of locked. **OPEN** Database has a global status of open. **STA** Database has a global status of started. **STOACC** Database has a global status of stopped for access. **STOSCHD** Database has a global status of stopped. **STOUPDS** Database has a global status of stopped for updates.
TMAC	TimeAccess	TIMESTAMP	N/A	The time that the resource was last accessed. The last access time is retained across warm start, emergency restart, export, and import. The updating of the last access time is not logged. After a restart, the last access time reflects the time that is recorded in the restart checkpoint log records. For a database resource, the following actions update the last access time: • Database is accessed by an application program. • CREATE command references the resource as the model. For HALDB and DEDB databases, the last access time stamp is returned in the HALDB partition entry or the AREA entry. The last access time stamp is left blank in the HALDB master and the DEDB entries.

Table 19. Output fields for QUERY DB command (continued)

Short label	Long label	Keyword	Scope	Meaning
TMCR	TimeCreate	TIMESTAMP	N/A	The time the resource was created. This is the result of a CREATE DB command, IMPORT command that creates the database, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMIM	TimeImport	TIMESTAMP	N/A	The time that the resource was last imported, if applicable. The import time is retained across warm start and emergency restart.
TMUP	TimeUpdate	TIMESTAMP	N/A	The time that the resource was updated as a result of an UPDATE DB command that changes a definitional attribute. This includes attributes set by the SET keyword, except for SET(LOCK(OFF/ON)). The update time is retained across warm start, emergency restart, export, and import.
TYP	Type	N/A	N/A	The type of the database. • AREA - indicates the response line is for a DEDB area • blank - if database status is NOTINIT • DEDB - indicates the database is a DEDB • DL/I - indicates the database is a full function non-partitioned database • MSNR - indicates the database is an MSDB non-related database • MSRD - indicates the database is an MSDB related dynamic database • MSRF - indicates the database is an MSDB related fixed database • PART - indicates the database is a HALDB partition • PHDAM - indicates the database is the master of a partitioned HDAM database • PHIDAM - indicates the database is the master of a Partitioned HIDAM database • PSINDEX - indicates the database is the master of a partitioned secondary index database
WRK	Work	WORK		Work is in progress for the database or one of its associated resources. Work in progress can be one of the following: **AREA OPEN** Area associated with the FP DEDB is open. **DB STOP ACCESS IN PROGRESS** A /DBRECOVERY or UPDATE DB STOP(ACCESS) command to stop database access is in progress for a database. This takes the database offline. **DB STOP UPDATES IN PROGRESS** A /DBDUMP or UPDATE DB STOP(UPDATES) command to stop database updates is in progress for the database. **IN USE** Database is in use. **RECOVER CMD ACTIVE** A /RECOVER START command is in progress to recover one or more databases with the database recovery services.

QUERY DB status

The following table lists local area status conditions that can be returned when SHOW(STATUS) is specified.

Table 20. Local status conditions for QUERY DB

Status	Meaning
ALLOCF	Database has an allocation failure.
ALLOCS	Database is allocated successfully.
BACKOUT	Incomplete backout exists for the database that prevents the use of the database.
EEQE	One or more extended error queue elements exist for the database.
IC	Area image copy is active.
LOCK	Database is locked locally.
NONE	Database has no global status.
NOTINIT-*xx-reason*	Database is not initialized and therefore cannot be used. NOTINIT is displayed in the format NOTINIT-*xx-reason*, where *xx* is the reason code that identifies the unique location in one module where this reason code is set. NOTINIT-00 indicates that the reason is unknown. Action: 1. DFSDDIR MACRO defines each reason code that might be set in the database bad reason code (field DDIRBADR) and identifies the module that sets it. *reason* explains the reason code *xx* in abbreviated text format up to 13 characters.
NOTOPEN	Area or database is not open.
OFR	Area or database has online forward recovery in progress to bring it up to current tracking level.
OLR	Database partition has an online reorganization in progress. The QUERY DB STATUS(OLR) command displays the status of all databases with HALDB OLR in progress. Rate information is not returned. Instead, the rate information is returned on the QUERY OLREORG command output.
OPEN	Area or database is open. Not maintained in the RM resource structure.
PRELOAD	Area is defined to be preloaded.
PREOPEN	Area is defined to be preopened.
QUIESCED	Area or database is currently quiesced by a previous UPDATE DB START(QUIESCE) or UPDATE AREA START(QUIESCE) command.
QUIESCING	Area or database is currently undergoing quiesce by a previous UPDATE DB START(QUIESCE) or UPDATE AREA START(QUIESCE) command.
RECALL	Area or database recall is in progress.
RECOVINP	Area or database recovery is in progress.
RECOVNEEDED	Area needs recovery.
RNL	Randomizer is not loaded for the DEDB database.

Table 20. Local status conditions for QUERY DB (continued)

Status	Meaning
STOACC	Database is stopped for access locally and is offline. The SHOW(STATUS) keyword on the QUERY DB command returns a status of STOACC for a DEDB. A status of STOACC indicates that the DEDB is stopped from further access because of a prior /DBR DB or UPDATE DB STOP(ACCESS) command.
STOPPED	Area is stopped locally or globally.
STOSCHD	Database is stopped locally.
STOUPDS	Database is stopped for updates locally.
VSO	Area is a VSO area.

Table 21. Reason information for NOTINIT-xx-reason status

Reason	Meaning
ALIAS	Alias name error.
BLDL	BLDL miscellaneous error trying to build ACBLIB directory.
DBINIT	Database initialization failed.
DMBINCOMPTBL	DMB incompatibility.
DMBLEVEL	The IMS release level at which this DMB is generated using ACBGEN does not match the IMS release level of this IMS. Perform a DBDGEN, PSBGEN, ACBGEN, and ACBLIB online change as needed to generate this DMB at the correct IMS release level. Action: 4.
DMBNAME	DMB name missing.
DMBPOOL	DMB pool shortage.
DMBPOOLDELETE	DMB delete from DMB pool failed.
DUPLICATEAREA	Duplicate area name found in this DEDB. Action: 4.
EOD	EOD marker found before DMB.
FPDB	The database is a Fast Path MSDB or DEDB that is defined in a non-Fast Path (FP=N) system. The database cannot be used.
FPRESTART	A Fast Path error occurred during restart. Action: 1.
LOADCOMPRESS	An error occurred loading the compression routine.
LOADRANDOMIZE	An error occurred loading the randomizer routine.
MAXDBEXCEEDED	Database exceeded maximum database limit of 32767. The database cannot be used. The database can be recovered at IMS cold start from RDDS auto import if the total number of databases in a local system is less than 32767.
MOLCCOMMIT	Member OLC COMMIT MEMBER failed.
MSDBCHANGE	MSDB added or changed by online change, which is not allowed. Action: 4.
MSDBLEVEL	MSDB level of MSDB in ACBLIB is incorrect. Action: 4.
NODB	No database DDIR control block exists. Action: 5.
NODMB	No database DMB exists in ACBLIB. For a Fast Path database, this could be an MSDB or a DEDB. Action: 2.
NOMSDB	No MSDBs defined in system.
NOSHRINDXDDIR	No shared index database DDIR control block. Action: 5.

Table 21. Reason information for NOTINIT-xx-reason status (continued)

Reason	Meaning
NOSHRINDXDMB	No DMB in ACBLIB for shared index. Action: 2.
NOTDMB	Not a DMB. A PSB by the same name as the database is defined in ACBLIB instead of a DMB. If this resource should be a program, create the program with a CREATE PGM command. If this resource should be a database, perform a DBDGEN, PSBGEN, ACBGEN, and ACBLIB online change to define this resource as a DMB instead of a PSB. Action: 4.
PARTBUILD	Partition build failed.
PSBINCOMPTBL	PSB incompatibility.
RSCNEEDOLC	A Fast Path DEDB database is created with the IMPORT DEFN command. The DEDB database cannot be brought online for use with the IMPORT DEFN command. Perform ACBLIB online change to bring Fast Path DEDB database online. Action 2.
RSCNEEDSTADB	A full-function HALDB is created with the IMPORT DEFN command. HALDB master is created. To create HALDB partition databases, issue either the /START DB command or the UPDATE DB command on the HALDB master database.
SEGMENTNUM	Segment number error. ACBGEN error, more than 127 segments. Action: 4.
WRONGDMB	Wrong DMB.
WRONGPSB	Wrong PSB.

Note: Actions that can be taken to initialize the database are:

1. Call IBM.
2. Perform ACBLIB online change to add the DMB to ACBLIB.
3. Perform ACBLIB online change to add the PSB to ACBLIB.
4. Perform ACBLIB online change to correct PSBs or DMBs.
5. Perform MODBLKS online change or issue the CREATE DB command to create the database.

How the SHOW keyword on QUERY DB determines the type of output

Some examples of how the SHOW keyword determines the type of output returned on the QUERY DB command are provided in the following table.

Table 22. How the SHOW keyword on QUERY DB determines the type of output

Form of SHOW keyword used	Type of output returned
SHOW(LOCAL)	Only those output fields that are local to an IMS system. SHOW(ALL,LOCAL) provides the same output.
SHOW(GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM. SHOW(ALL,GLOBAL) provides the same output.
SHOW(ALL)	All of the output fields for those fields that have both local and global data. Both values are returned in the output.
SHOW(STATUS,GLOBAL)	Only global STATUS values.
SHOW(STATUS,LOCAL)	Only local STATUS values.
SHOW(STATUS)	Both local and global STATUS values.

Table 22. How the SHOW keyword on QUERY DB determines the type of output (continued)

Form of SHOW keyword used	Type of output returned
SHOW(ALL,GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM. SHOW(GLOBAL) provides the same output.
SHOW(ALL,LOCAL)	Only those output fields that are local to an IMS system. SHOW(LOCAL) provides the same output.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY DB command. The OM return and reason codes that might be returned as a result of the QUERY DB command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 23. Return and reason codes for the QUERY DB command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY DB command completed successfully.
X'00000004'	X'00001010'	No resources were found to be returned. The resource names specified might be invalid, or there were no resources that match the filter specified, or there were no resources that had work to display for the SHOW(WORK) specified.
X'00000008'	X'00002014'	The QUERY DB command is not processed because an invalid character is found in the database name parameter.
X'00000008'	X'00002040'	More than one filter or keyword value is specified on the QUERY DB command. Either more than one keyword or an invalid combination of filters was specified. Check the input command and reenter the correct combinations.
X'0000000C'	X'00003000'	The QUERY DB command was successful for at least one resource name. The QUERY DB command was not successful for one or more resource names. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY DB command are listed in the QUERY DB completion code table.
X'0000000C'	X'00003004'	The QUERY DB command was not successful for all the resource names specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY DB command are listed in the QUERY DB completion code table.
X'00000014'	X'00005004'	The QUERY DB command processing terminated as a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The QUERY DB command processing terminated because of an internal error.

The following table includes an explanation of the completion codes. Errors unique to the processing of QUERY DB command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 24. Completion codes for the QUERY DB command

Completion code	Completion code text	Meaning
0		The QUERY DB command completed successfully for the resource.
10	RESOURCE NOT FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. Confirm the correct spelling of the resource name specified on the command.

Examples

The following are examples of the QUERY DB command:

Example 1 for QUERY DB command

TSO SPOC input:

```
QUERY DB NAME(BANKATMS,DEDBJN21,DBHDOK01) SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
DBName   AreaName MbrName    CC TYPE     LAcc LRsdnt LclStat
BANKATMS          IMS1        0          EXCL N      NOTINIT-1E-NODMB,NOTOPEN
DBHDOK01          IMS1        0 DL/I     UPD  N      NOTOPEN
DEDBJN21          IMS1        0 DEDB     UPD  Y      NOTOPEN
DEDBJN21 DB21AR0  IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR1  IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR10 IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR11 IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR2  IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR3  IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR4  IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR5  IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR6  IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR7  IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR8  IMS1        0 AREA                 NOTOPEN
DEDBJN21 DB21AR9  IMS1        0 AREA                 NOTOPEN

(scrolled right to screen 2)
DBName   AreaName MbrName  ModelName  ModelType TimeCreate
BANKATMS          IMS1                         2006.254 09:15:27.24
DBHDOK01          IMS1                         2006.254 09:15:27.24
DEDBJN21          IMS1                         2006.254 09:15:27.24
DEDBJN21 DB21AR0  IMS1
DEDBJN21 DB21AR1  IMS1
DEDBJN21 DB21AR10 IMS1
DEDBJN21 DB21AR11 IMS1
DEDBJN21 DB21AR2  IMS1
DEDBJN21 DB21AR3  IMS1
DEDBJN21 DB21AR4  IMS1
DEDBJN21 DB21AR5  IMS1
DEDBJN21 DB21AR6  IMS1
DEDBJN21 DB21AR7  IMS1
DEDBJN21 DB21AR8  IMS1
DEDBJN21 DB21AR9  IMS1

(scrolled right to screen 3)
DBName   AreaName MbrName  TimeUpdate TimeAccess          TimeImport
BANKATMS          IMS1
DBHDOK01          IMS1
```

```
DEDBJN21          IMS1
DEDBJN21 DB21AR0  IMS1
DEDBJN21 DB21AR1  IMS1
DEDBJN21 DB21AR10 IMS1
DEDBJN21 DB21AR11 IMS1
DEDBJN21 DB21AR2  IMS1
DEDBJN21 DB21AR3  IMS1
DEDBJN21 DB21AR4  IMS1
DEDBJN21 DB21AR5  IMS1
DEDBJN21 DB21AR6  IMS1
DEDBJN21 DB21AR7  IMS1
DEDBJN21 DB21AR8  IMS1
DEDBJN21 DB21AR9  IMS1

(scrolled right to screen 4)
DBName   AreaName MbrName  TimeAccess           TimeImport           DefnType
BANKATMS          IMS1                                               MODBLKS
DBHDOK01          IMS1                                               MODBLKS
DEDBJN21          IMS1                                               MODBLKS
DEDBJN21 DB21AR0  IMS1
DEDBJN21 DB21AR1  IMS1
DEDBJN21 DB21AR10 IMS1
DEDBJN21 DB21AR11 IMS1
DEDBJN21 DB21AR2  IMS1
DEDBJN21 DB21AR3  IMS1
DEDBJN21 DB21AR4  IMS1
DEDBJN21 DB21AR5  IMS1
DEDBJN21 DB21AR6  IMS1
DEDBJN21 DB21AR7  IMS1
DEDBJN21 DB21AR8  IMS1
DEDBJN21 DB21AR9  IMS1
```

OM API input:

```
CMD(QUERY DB NAME(BANKATMS,DEDBJN21,DBHDOK01) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.254 17:39:17.155247</statime>
<stotime>2006.254 17:39:17.155646</stotime>
<staseq>BF641524199AFCC5</staseq>
<stoseq>BF64152419B3E645</stoseq>
<rqsttkn1>USRT011 10103917</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>DB              </kwd>
<input>QUERY DB NAME(BANKATMS,DEDBJN21,DBHDOK01) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="4"
scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
```

```
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RSDNT" llbl="Rsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LRSDNT" llbl="LRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MDLN" llbl="ModelName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MDLT" llbl="ModelType" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMUP" llbl="TimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMAC" llbl="TimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="TimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="DFNT" llbl="DefnType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) TYP(        ) LACC(EXCL)
 LSTT(NOTINIT-1E-NODMB,NOTOPEN) DFNT(MODBLKS) MDLN(            ) MDLT()
 LRSDNT(N) TMCR(2006.254 09:15:27.24) TMUP(                    ) TMIM(
                  ) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) TYP(DEDB    ) LACC(UPD)
 LSTT(NOTOPEN) DFNT(MODBLKS) MDLN(            ) MDLT() LRSDNT(Y)
 TMCR(2006.254 09:15:27.24) TMUP(                    ) TMIM(
         ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR0 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR1 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR2 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR3 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR4 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR5 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR6 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR7 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR8 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR9 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR10) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR11) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) TMAC(                    ) </rsp>
<rsp>DB(DBHDOK01) MBR(IMS1    ) CC(   0) TYP(DL/I    ) LACC(UPD)
 LSTT(NOTOPEN) DFNT(MODBLKS) MDLN(            ) MDLT() LRSDNT(N)
 TMCR(2006.254 09:15:27.24) TMUP(                    ) TMIM(
         ) TMAC(                   ) </rsp>
</cmdrspdata>
 </imsout>
```

Explanation: The QUERY DB command is specified with SHOW keywords to display the resident attribute and the database status. All definition and status

information is returned for databases BANKATMS, DEDBJN21 and DBHDOK01 from IMS1. All of the database output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The database name, area name, and member name that built the line of output are displayed on every screen. Database BANKATMS has a status of NOTINIT-1E-NODMB, which means there is no DMB in ACBLIB for BANKATMS and BANKATMS cannot be used. DEDBJN21 is a DEDB and the area information is also returned, along with the DEDB information. No model information is returned, because the database was loaded from MODBLKS and not created from a model. The time stamps for the DEDB areas are not maintained and so it is not returned.

Example 2 for QUERY DB command

TSO SPOC input:

```
QUERY DB NAME(BE3PARTS,CUSTDB) SHOW(PGM)
```

TSO SPOC output:

```
DBName   MbrName    CC TYPE     PgmName
BE3PARTS IMS1        0 DL/I     BE3PARTS
BE3PARTS IMS1        0 DL/I     PE4COINQ
BE3PARTS IMS1        0 DL/I     PE4CORDR
CUSTDB   IMS1        0 DEDB     CUSTDB
CUSTDB   IMS1        0 DEDB     PSBOS
CUSTDB   IMS1        0 DEDB     PSBPA
```

OM API input:

```
CMD(QUERY DB NAME(BE3PARTS,CUSTDB) SHOW(PGM))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.254 17:57:37.630168</statime>
<stotime>2006.254 17:57:37.630961</stotime>
<staseq>BF64193D983D8FC5</staseq>
<stoseq>BF64193D986F1645</stoseq>
<rqsttkn1>USRT011 10105737</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>DB              </kwd>
<input>QRY DB NAME(BE3PARTS,CUSTDB) SHOW(PGM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
```

```
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="PGM" llbl="PgmName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) TYP(DL/I    ) PGM(BE3PARTS)  </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) TYP(DL/I    ) PGM(PE4COINQ)  </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) TYP(DL/I    ) PGM(PE4CORDR)  </rsp>
<rsp>DB(CUSTDB  ) MBR(IMS1    ) CC(   0) TYP(DEDB    ) PGM(CUSTDB  )  </rsp>
<rsp>DB(CUSTDB  ) MBR(IMS1    ) CC(   0) TYP(DEDB    ) PGM(PSBOS   )  </rsp>
<rsp>DB(CUSTDB  ) MBR(IMS1    ) CC(   0) TYP(DEDB    ) PGM(PSBPA   )  </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Database BE3PARTS has programs BE3PARTS, PE4COINQ, and PE4CORDR referring to it. Database CUSTDB has programs CUSTDB, PSBOS, and PSBPA referring to it.

Example 3 for QUERY DB command

TSO SPOC input:

```
QRY DB NAME(DEDBRN01,DX41M401,DB000001) SHOW(RESIDENT,STATUS)
```

TSO SPOC output:

```
Response for: QRY DB NAME(DEDBRN01,DX41M401,DB000001) SHOW(RESIDENT,STATUS)
DBName   AreaName MbrName    CC TYPE     Rsdnt LRsdnt LclStat
DB000001          IMS1        0          Y     N      NOTINIT-34-NODMB,NOTOPEN
DEDBRN01          IMS1        0 DEDB           Y      NOTOPEN
DEDBRN01 DEDB01D1 IMS1        0 AREA                  NOTOPEN
DX41M401          IMS1        0 DL/I           N      NOTOPEN
```

OM API input:

```
CMD(QUERY DB NAME(DEDBRN01,DX41M401,DB000001) SHOW(RESIDENT,STATUS))
```

OM API output:

```
<imsout>
 <ctl>
 <omname>OM1OM   </omname>
 <omvsn>1.3.0</omvsn>
 <xmlvsn>20  </xmlvsn>
 <statime>2006.286 17:15:56.248881</statime>
 <stotime>2006.286 17:15:56.249254</stotime>
<staseq>BF8C4B9AD7731208</staseq>
<stoseq>BF8C4B9AD78A67C8</stoseq>
<rqsttkn1>USRT011 10101556</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>DB              </kwd>
<input>QRY DB NAME(DEDBRN01,DX41M401,DB000001) SHOW(RESIDENT,STATUS)
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="4"
```

```
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RSDNT" llbl="Rsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LRSDNT" llbl="LRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(DEDBRN01) MBR(IMS1) CC(   0) TYP(DEDB    ) LSTT(NOTOPEN)
 LRSDNT(Y) </rsp>
<rsp>DB(DEDBRN01) AREA(DEDB01D1) MBR(IMS1) CC(   0) TYP(AREA    )
 LSTT(NOTOPEN) </rsp>
<rsp>DB(DX41M401) MBR(IMS1) CC(   0) TYP(DL/I    ) LSTT(NOTOPEN)
 LRSDNT(N) </rsp>
<rsp>DB(DB000001) MBR(IMS1) CC(   0) TYP(        )
 LSTT(NOTINIT-34-NODMB,NOTOPEN) RSDNT(Y) LRSDNT(N) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY DB command is specified with SHOW keywords to display the resident attribute and the database status for a few different types of databases. DEDBRN01 is a DEDB. DEDBs are always resident and defined with RESIDENT(Y). DX41M401 is a DLI database. DX41M401 is defined as Resident(N) and its local runtime resident value is RESIDENT(N). Since Rsdnt and LRsdnt values of database DX41M401 are the same, only LRsdnt value is displayed. DB000001 is a new database created by a CREATE DB command. DB000001 shows a status of NOTINIT-34-NODMB, which means it has no DMB defined in ACBLIB and cannot be used. Since there is no DMB, the database type is shown as blanks, because IMS does not know what type of database DB000001 is intended to be. DB000001 is defined as RESIDENT(Y), but since the resident option will not take effect until the next IMS restart, the local resident value is N. If all of the databases being displayed have a LRsdnt (Local Resident) value that is the same as the Rsdnt (Definitional Resident) value, the QRY DB output displays the LRsdnt header and value, but not the Rsdnt header and value. Since database DB000001 is defined with RESIDENT(Y) but its local runtime resident value is N, both Rsdnt and LRsdnt columns are displayed in the QUERY DB output.

Example 4 for QUERY DB command

TSO SPOC input:

```
QRY DB NAME(*) SHOW(STATUS) STATUS(QUIESCING)
```

TSO SPOC output:

```
DBName    MbrName    CC TYPE     LclStat
DBXYZ     IM03        0 DLI      ALLOCS,OPEN,QUIESCING
KCIRDBAZ  IM03        0 DLI      ALLOCS,OPEN,QUIESCING
GONLXYZ   IM03        0 DLI      ALLOCS,OPEN,QUIESCING
```

Explanation: This example is a query of the databases that have a status of quiescing.

Example 5 for QUERY DB command

TSO SPOC input:

```
QUERY DB NAME(DEDBJ001) SHOW(ACCTYPE,STATUS)
```

TSO SPOC output:

```
DBName   AreaName MbrName    CC TYPE     LAcc LclStat
DEDBJ001          IMS1        0 DEDB     UPD  OPEN
DEDBJ001 D0010001 IMS1        0 AREA     READ OPEN,PREOPEN
DEDBJ001 D0010002 IMS1        0 AREA     BRWS OPEN,PREOPEN
DEDBJ001 D0010003 IMS1        0 AREA          OPEN,PREOPEN
DEDBJ001 D0010004 IMS1        0 AREA     UPD  OPEN,PREOPEN
DEDBJ001 D0010005 IMS1        0 AREA          OPEN,PREOPEN
```

OM API input:

```
CMD(QUERY DB NAME(DEDBJ001) SHOW(ACCTYPE,STATUS))
```

OM API output:

```
<imsout>
 <ctl>
 <omname>OM1OM   </omname>
 <omvsn>1.5.0</omvsn>
 <xmlvsn>20  </xmlvsn>
 <statime>2011.188 23:38:28.734794</statime>
 <stotime>2011.188 23:38:28.735444</stotime>
<staseq>C80937F889B4AC94</staseq>
<stoseq>C80937F889DD4554</stoseq>
<rqsttkn1>USRT011 10163828</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>DB              </kwd>
<input>QRY DB NAME(DEDBJ001) SHOW(ACCTYPE,STATUS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(DEDBJ001) MBR(IMS1    ) CC(   0) TYP(DEDB    ) LACC(UPD)
 LSTT(OPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010001 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LACC(READ)LSTT(OPEN,PREOPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010002 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LACC(BRWS) LSTT(OPEN,PREOPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010003 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
```

```
 LSTT(OPEN,PREOPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010004 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LACC(UPD)LSTT(OPEN,PREOPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010005 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(OPEN,PREOPEN) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: DEDB DEDBJ001 has database access of UPD (update). Area access for areas D0010001, D00010002, and D0010004 has been changed with the command; that is, area D0010001 has area access of READ (read), area D0010002 has area access of BRWS (read only), and area D0010004 has area access of UPD (update). Areas D0010003 and D0010005 inherit DEDBJ001 database access of UPD (update).

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY DB command (Commands)

/DISPLAY DBD command (Commands)

/DISPLAY MODIFY command (Commands)

/DISPLAY STATUS command (Commands)

Command keywords and their synonyms (Commands)

QUERY DBDESC command

Use the QUERY DBDESC command to query information about database descriptors. A descriptor is a model that can be used to create descriptors or resources.

Subsections:

- "Environment"
- "Syntax" on page 72
- "Keywords" on page 72
- "Usage notes" on page 73
- "Output fields" on page 73
- "Return, reason, and completion codes" on page 76
- "Examples" on page 77

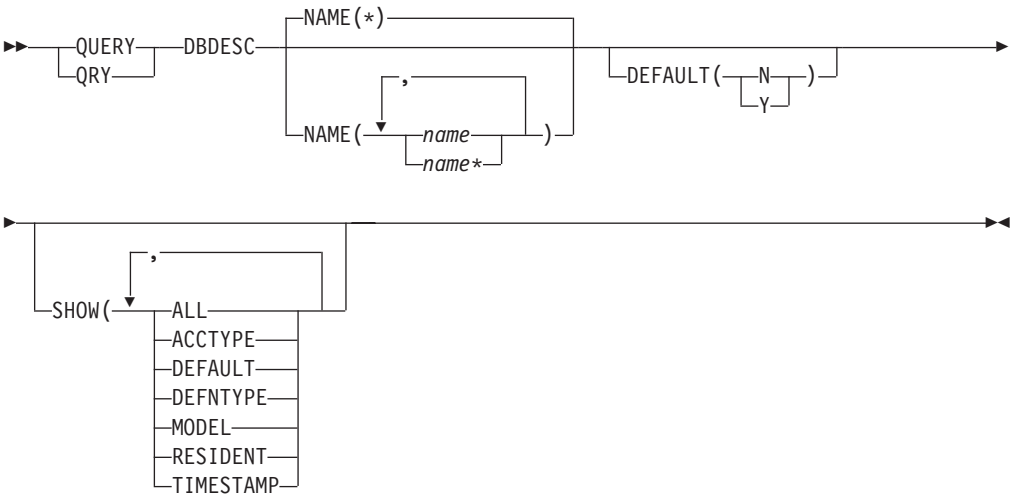
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 25. Valid environments for the QUERY DBDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY DBDESC	X	X	
NAME	X		X
DEFAULT	X		X
SHOW	X		X

Syntax



Keywords

The following keywords are valid for the QUERY DBDESC command:

DEFAULT()
Specifies which descriptor or descriptors to display.

N Displays all the database descriptors that are not the default.

Y Displays the default database descriptor.

NAME()
Specifies the 1-8 character name of the database descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all database descriptors.

SHOW()
Specifies the database descriptor output fields to be returned. The database descriptor name is always returned, along with the name of the IMS that created the output for the database descriptor and the completion code. The filters supported with the SHOW keyword are:

ALL
Returns all information about the database descriptor itself.

ACCTYPE
Type of access to the database created using this database descriptor.

DEFAULT
Default descriptor option.

DEFNTYPE
Definition type. This is how the descriptor was defined.

MODEL
The model name and model type used to create this descriptor. The model name and model type is blank for the IMS-defined descriptor DFSDSDB1. The CREATE command specified without the LIKE keyword creates a

descriptor using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSDB1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor using a model. The descriptor is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify descriptors that were created with the same model. The model name and model type of a descriptor are exported and imported. The IMPORT command does not use the model name and model type when creating a descriptor.

RESIDENT
Specifies the resident option. The value is always RESIDENT(Y) for Fast Path DEDBs. The RESIDENT(Y) option takes effect at the next restart, unless the database descriptor was created or updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS) time, and import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

Usage notes

This command can be issued only through the Operations Manager API. This command applies to DB/DC and DBCTL systems. This command is allowed on XRF alternate and RSR tracker systems. The QUERY DBDESC command is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

Output fields

The following table shows the QUERY DBDESC output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 26. Output fields for the QUERY DBDESC command

Short label	Keyword	Meaning
CC	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified descriptor. The completion code is always returned. Refer to the return, reason, and completion codes for QUERY DBDESC.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DESC	DBDESC	Database descriptor name.
DFLT	DBDESC	Default descriptor (Y) or not (N). **N** The descriptor is not the default. **Y** The descriptor is the default. When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a database descriptor called DFSDSDB1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor.
DFNT	DEFNTYPE	Definition type, which can be one of the following: **CREATE** Defined by a CREATE command. **IMPORT** Defined by an IMPORT command. The DEFNTYPE is not changed if the descriptor or resource is updated with an UPDATE command. **IMS** Defined by IMS. DFSDSDB1 is an IMS-defined database descriptor containing the default database descriptor values.

Table 26. Output fields for the QUERY DBDESC command (continued)

Short label	Keyword	Meaning
LACC	ACCTYPE	Type of access to database descriptor which can be one of the following: **BRWS** The database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data since a sharing IMS subsystem could be updating the database. The database is opened for read-only processing. **EXCL** The named database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC. **READ** The database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. Access type READ differs from access type BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing. **UPD** The database is available for update as well as read processing in the IMS subsystem.
MBR	N/A	IMSplex member that built the output line. The IMS identifier of the IMS that built the output. The IMS identifier is always returned.
MDLN	MODEL	Model name. Name of the resource or descriptor used as a model to create this descriptor. DFSDSDB1 is the IMS descriptor name for databases.
MDLT	MODEL	Model type, either RSC or DESC. RSC means the descriptor was created using another resource as a model. DESC means the descriptor was created using a descriptor as the model.

Table 26. Output fields for the QUERY DBDESC command (continued)

Short label	Keyword	Meaning
RSDNT	RESIDENT	Resident option. Indicates whether the database descriptor DMB is to reside in local storage at the next IMS restart. This value is only returned if the resident option definition is different from the resident option local runtime value. **N** The DMB associated with the named database descriptor resource is not made resident in storage. The DMB is loaded at scheduling time. **Y** The DMB associated with the named database descriptor resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the DMB and initializes it. A resident database descriptor is accessed from local storage, which eliminates I/O to the ACBLIB. In an online environment, the DMB control blocks are stored in the ACBLIB. If the DLI/SAS address space exists, DLI/SAS loads the DMB, otherwise, it is the IMS control region that loads it. This makes the DMB dependent on the existence of the corresponding database descriptor resource.
TMAC	TIMESTAMP	The time that the descriptor was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. For a database descriptor, when the CREATE command references the descriptor as the model, the last access time is updated.
TMCR	TIMESTAMP	The time the descriptor was created. This is the result of a CREATE DB command, IMPORT command that creates the database descriptor, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMIM	TIMESTAMP	The time that the descriptor was last imported, if applicable. The import time is retained across warm start and emergency restart.
TMUP	TIMESTAMP	The time that the descriptor was updated as a result of an UPDATE DB command that changes a runtime resource definition attribute. This includes attributes set by the SET keyword, except for SET(LOCK(OFF/ON)). The update time is retained across warm start, emergency restart, EXPORT and IMPORT.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 27. Return and reason codes for the QUERY DBDESC command

Return code	Reason code	Meaning
X'00000004'	X'00001010'	No descriptors were found to be returned. The descriptor name(s) specified may be invalid, or there were no descriptors that match the filter specified.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 28. Completion codes for the QUERY DBDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for database or database descriptor.
10	NO RESOURCES FOUND	Database descriptor name is invalid, or the wildcard parameter specified does not match any database descriptor names.

Examples

The following are examples of the QUERY DBDESC command:

Example 1 for QUERY DBDESC command

TSO SPOC input:

```
QRY DBDESC SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
DescName MbrName    CC LAcc Rsdnt Dflt ModelName    ModelType TimeCreate
DFSDSDB1 IMS1        0 UPD  N     Y                            2006.286 09:14:53.39
(scrolled to the right screen 2)
DescName MbrName  TimeUpdate           TimeAccess           TimeImport>
DFSDSDB1 IMS1
(scrolled to the right screen 3)
DescName MbrName  TimeAccess           TimeImport           DefnType
DFSDSDB1 IMS1
```

OM API input:

```
CMD(QUERY DBDESC SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.286 17:02:01.076428</statime>
<stotime>2006.286 17:02:01.077017</stotime>
<staseq>BF8C487E5BECCB4C</staseq>
```

```
<stoseq>BF8C487E5C11950C</stoseq>
<rqsttkn1>USRT011 10100201</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>DBDESC          </kwd>
<input>QRY DBDESC SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RSDNT" llbl="Rsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DFLT" llbl="Dflt" scope="LCL" sort="n" key="0"
 scroll="yes"len="1" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MDLN" llbl="ModelName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MDLT" llbl="ModelType" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMUP" llbl="TimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMAC" llbl="TimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="TimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="DFNT" llbl="DefnType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DFSDSDB1) MBR(IMS1) CC(   0) LACC(UPD) DFNT(IMS) MDLN(
     ) MDLT() RSDNT(N) TMCR(2006.286 09:14:53.39) TMUP(
     ) TMIM(                    ) TMAC(                    ) DFLT(Y)
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The database descriptors are displayed. There is one database descriptor in the system: the IMS-defined descriptor DFSDSDB1. All output fields are returned for the descriptor. All of the database descriptor output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The database descriptor name and member name that built the line of output are displayed on every screen. The time the DFSDSDB1 descriptor was created was at IMS cold start time. IMS-defined descriptors display a blank ModelName and ModelType, since no model is used to create them. The Dflt column shows a value of Y, which indicates that DFSDSDB1 is defined as the default database descriptor.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY IMS command

Use the QUERY IMS command, which is a type-2 command, to display status and attributes information from IMS.

The information that is returned to the user is the IMS status that can be modified with the UPDATE IMS command. The QUERY MEMBER command is used to return other IMS status.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 80
- "Output fields" on page 80
- "Return, reason, and completion codes" on page 82
- "Examples" on page 83

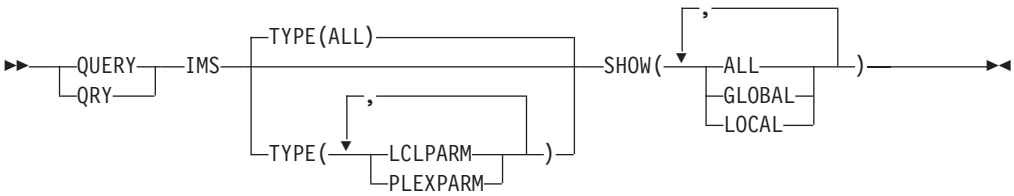
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY IMS command and keywords can be issued.

Table 29. Valid environments for QUERY IMS command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUERY IMS	X	X	X
SHOW	X	X	X
TYPE	X	X	X

Syntax



Keywords

The following keywords are valid for the QUERY IMS command:

SHOW()
(Required) - Specifies the output fields to be returned.

ALL
Returns all possible output fields for the data specified in the TYPE parameter.

GLOBAL
Returns the global values for the specified TYPE options.

LOCAL
Returns the local values for the specified TYPE options.

TYPE()
Specifies the data to be returned.

ALL
Returns data for all possible types. ALL is the default.

LCLPARM
Displays the local LCLPARM values at the current IMS. Information is returned based on the SHOW options specified.

You cannot specify SHOW(GLOBAL) with TYPE(LCLPARM). SHOW(ALL) and SHOW(LOCAL) return the same local parameter information.

PLEXPARM
Displays the global PLEXPARM values at the current IMS.

If global values are requested, the values from the global PLEXPARM entry in RM are returned by the IMS command master. If local values are requested, the values from each IMS that processes the command are returned.

Usage notes

This command can be specified only through the Operations Manager API.

Output fields

The output provided by QUERY IMS is described in the following table. Specifying SHOW parameter options determines what is returned in the output. Specifying SHOW(ALL) returns all output fields.

The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 30. QUERY IMS output fields

Short label	Keyword	Scope	Meaning
BMP	LCLPARM	LCL	Amount of time that IMS waits before lock requests for BMP regions are timed out. BMP regions include IMS BMP and JBP regions. The value, which represents time in seconds, can range from 1 to 32767.

Table 30. QUERY IMS output fields (continued)

Short label	Keyword	Scope	Meaning
BMPOPT	LCLPARM	LCL	Indicates whether IMS ends a timed-out task abnormally (ABEND) or returns a status code to the application (STATUS).
CC	n/a	n/a	Completion code, which indicates whether IMS was able to process the command for the specified resource. The completion code is always returned.
CCTXT	*error*	LCL	Text returned with the completion code to provide additional information. It can include a return code from a service. Completion code text is returned only if the completion code is nonzero.
FP64STAT	LCLPARM	LCL	Fast Path 64-bit buffer usage statistics per unit of work for dependent regions. This field is displayed only when Fast Path 64-bit buffer manager is enabled. Fast Path 64-bit buffer usage statistics are recorded in X'5945' log records. The values include: **N** Does not write Fast Path 64-bit buffer usage statistics per unit of work for dependent regions in X'5945' log records to OLDS. **Y** Writes Fast Path 64-bit buffer usage per unit of work for dependent regions in X'5945' log records to OLDS.
GSTSAREA	PLEXPARM	LCL or GBL	The global status for areas. The values include: **Y** Global status is maintained for areas. **N** Global status is not maintained for areas. **NULL** Global status has not been set for areas.
GSTSDB	PLEXPARM	LCL or GBL	Global status for databases. The values include: **Y** Global status is maintained for databases. **N** Global status is not maintained for databases. **NULL** Global status has not been set for databases.
GSTSTRAN	PLEXPARM	LCL or GBL	Global status for transactions. The values include: **Y** Global status is maintained for transactions. **N** Global status is not maintained for transactions. **NULL** Global status has not been set for transactions.
MBR	n/a	n/a	The IMS identifier of the IMS that built the output. The IMS identifier is always returned.

Table 30. QUERY IMS output fields (continued)

Short label	Keyword	Scope	Meaning
MSG	LCLPARM	LCL	Amount of time that IMS waits before lock requests for MSG regions are timed out. MSG regions include IMS MPP, JMP, and IFP regions as well as DRA threads. The value, which represents time in seconds, can range from 1 to 32767.
MSGOPT	LCLPARM	LCL	Indicates whether IMS ends a timed-out task abnormally (ABEND) or returns a status code to the application (STATUS).

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMS command are standard for all commands that are entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 31. Return and reason codes for the QUERY IMS command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMS command completed successfully.
X'00000004'	X'00001000'	The QUERY IMS command is not processed on the IMS system because the IMS system is not the command master.
X'00000008'	X'00002040'	The QUERY IMS command is not processed because no filter or an invalid filter was specified for the SHOW keyword.
X'00000014'	X'00005004'	The QUERY IMS command is not processed because the DFSOCMD0 GETBUF storage could not be obtained.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains completion codes can be returned on a QUERY IMSPLEX command.

Table 32. Completion codes for the QUERY IMS command

Completion code	Meaning
0	The QUERY IMS command completed successfully.
50	The QUERY IMS command is not processed because CQS is not available.
51	The QUERY IMS command is not processed because there is no resource structure.
52	The QUERY IMS command is not processed because the resource structure is full.
53	The QUERY IMS command is not processed because RM is not available.

Table 32. Completion codes for the QUERY IMS command (continued)

Completion code	Meaning
54	The QUERY IMS command is not processed because SCI is not available.
90	The QUERY IMS command is not processed because of an IMS internal error.
94	The QUERY IMS command is not processed because of an RM error.
95	The QUERY IMS command is not processed because of an RM error.
98	The QUERY IMS command is not processed because of a CQS error.

Examples

The following are examples of the QUERY IMS command:

Example 1 for the QUERY IMS ***command***

TSO SPOC input:

```
QUERY IMS TYPE(LCLPARM) SHOW(ALL)
```

TSO SPOC output:

```
      MbrName    CC  FPBP64STAT
      IMS1        0  N
      IMS2        0  N
```

Example 2 for the QUERY IMS ***command***

TSO SPOC input:

```
QUERY IMS TYPE(LCLPARM) SHOW(LOCAL)
```

TSO SPOC output:

```
      MbrName    CC  FPBP64STAT
      IMS1        0  N
      IMS2        0  N
```

Example 3 for the QUERY IMS ***command***

TSO SPOC input:

```
QUERY IMS TYPE(LCLPARM) SHOW(ALL)
```

TSO SPOC output:

```
   MbrName    CC   MSG  MSGOPT   BMP  BMPOPT
   IMS1        0    40  ABEND     30  STATUS
   IMS2        0    40  ABEND     30  STATUS
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY IMSPLEX command

Use the QUERY IMSPLEX command, which is a type-2 command, to display information about one or more IMSplex members.

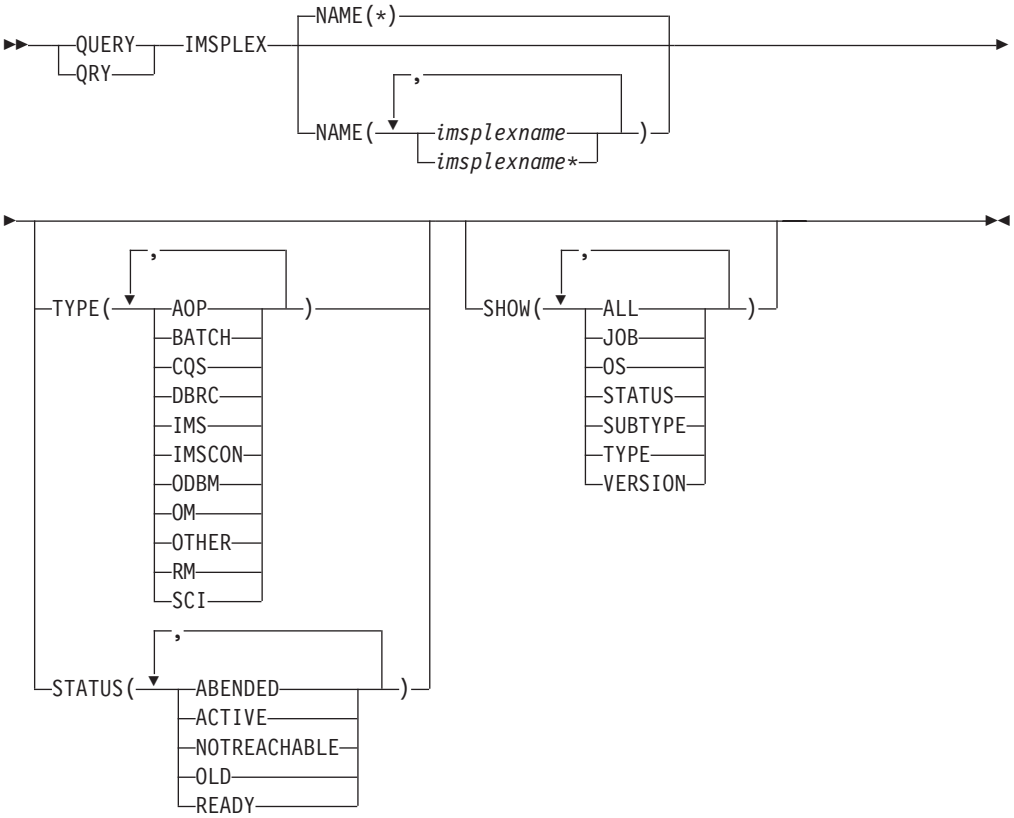
Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 86
- "Output fields" on page 88
- "Return, reason, and completion codes" on page 89
- "Examples" on page 90

Environment

The QUERY IMSPLEX command does not run in the address space of any IMS control or dependent region. QUERY IMSPLEX is processed in an OM command processing environment.

Syntax



Keywords

The following keywords are valid for the QUERY IMPSLEX command:

NAME()
Specifies the name of the IMSplexes for which member information is to be returned. The IMSplex name can be a generic parameter, to allow easy specification of a group of IMSplexes whose names match a generic parameter mask. For example, QUERY IMSPLEX NAME(*PLEX*).

You must include the prefix, CSL, to the name of the IMSplex for which you want member information. Add CSL in front of the IMSplex name that you specified in the IMSPLEX= parameter in the DFSCGxxx PROCLIB member. For example, if you specified IMSPLEX=PLEX1 in your DFSCGxxx member, you must specify QUERY IMSPLEX NAME(CSLPLEX1).

SHOW()
Specifies the output fields to be returned. If SHOW is not specified, only the IMSplex names, IMSplex member names, IMSplex member that builds the output line, and completion codes are returned. This provides a method for a system management application to obtain a list of all IMSplex member names that are currently known in the IMSplexes.

ALL
Returns all output fields.

JOB
Job name of the IMSplex member.

OS Name of the OS image on which the IMSplex member is executing.

STATUS
IMSplex member status.

SUBTYPE
IMSplex member subtype.

TYPE
IMSplex member type.

VERSION
IMSplex member version.

STATUS()
Displays IMSplex members that have at least one of the specified statuses.

ABENDED
IMSplex member has ended abnormally.

ACTIVE
IMSplex member is active.

NOTREACHABLE
The local SCI responsible for the member is not currently active. The status displayed is the current status for the member.

Note: The status output is NOT-REACHABLE (with a hyphen).

OLD
The SCI responsible for the member is not currently active. The status displayed is the last known status for the member. The actual status might be different.

READY
IMSplex member is ready to receive messages and requests that are routed to it by any method, including by TYPE.

TYPE()
Displays IMSplex members that possess at least one of the specified member types.

AOP
Automated Operator Program. Examples of AOPs are a SPOC application that an operator uses to interact with an IMSplex or a program that is monitoring an IMSplex.

BATCH
IMS batch job.

CQS
Common Queue Server address space.

DBRC
DBRC address space.

IMS
IMS region.

IMSCON
An address space that serves as an interface between IMS and a protocol that is not directly supported by IMS (for example, TCP/IP).

ODBM
Open Database Manager address space.

OM Operations Manager address space.

OTHER
Other non-IMS address space or job.

RM Resource Manager address space.

SCI
Structured Call Interface address space.

Usage notes

This command can be issued only through the OM API.

QUERY IMSPLEX status

The following table shows the possible IMSplex member status. The table contains information about status such as the STATUS keyword to specify to select members with the specified status, the status that is returned, and the meaning of the status.

Table 33. QUERY IMSPLEX status table

Status keyword	Status	Meaning
ABENDED	ABENDED	IMSplex member has abended.
ACTIVE	ACTIVE	IMSplex member is active.
NOTREACHABLE	NOT-REACHABLE	The local SCI responsible for the member is currently not active. The status displayed is the current status for the member.
OLD	OLD	The SCI responsible for the member is not currently active. The status displayed is the last known status for the member. The actual status might be different.

Table 33. QUERY IMSPLEX status table (continued)

Status keyword	Status	Meaning
READY	READY	IMSplex member is ready to receive messages and requests that are routed to it by any method, including by TYPE.

QUERY IMSPLEX types

The following table shows the possible IMSplex member types. The table contains information about member types such as the TYPE keyword to specify to select members with the specified type, the type that is returned, and the meaning of the member type.

Table 34. QUERY IMSPLEX member types

Type keyword	Member type	Meaning
AOP	aop	Automated Operator Program. An example of an AOP is a SPOC (Single Point of Control) that an operator uses to interact with the IMSplex. Another example of an AOP is a program that is monitoring the IMSplex.
BATCH	batch	IMS batch job.
CQS	cqs	Common Queue Server address space. CQS manages shared queues and may also manage resources on a resource structure.
DBRC	dbrc	DBRC address space.
IMS	ims	IMS region.
IMSCON	imscon	IMS connect. An address space that serves as an interface between IMS and a protocol that is not directly supported by IMS.
ODBM	odbm	Open Database Manager address space. ODBM provides distributed access to IMS database resources.
OM	om	Operations Manager address space. Operations Manager supports IMS operations in an IMSplex.
OTHER	other	Other non-IMS address space or job.
RM	rm	Resource Manager address space. Resource manager supports global resources in an IMSplex.
SCI	sci	Structured Call Interface address space.

QUERY IMSPLEX subtypes

The following table shows the possible IMSplex member subtypes. The table contains information about member types, the member subtypes associated with them, and the meaning of the member subtype. These are the only subtypes that are defined and used by members supplied by IMS.

Table 35. QUERY IMSPLEX member subtypes

Member type	Member subtype	Meaning
DBRC	DBRC group_id	DBRC instances that share the same RECON in an IMSplex environment.

Table 35. QUERY IMSPLEX member subtypes (continued)

Member type	Member subtype	Meaning
IMS	DBDC	IMS DB/DC address space.
IMS	DBCTL	IMS DBCTL address space. DBCTL supports database functions.
IMS	DCCTL	IMS DCCTL address space. DCCTL supports data communications functions.
IMS	FDBR	IMS Fast Database Recovery. An IMS control region that recovers database resources when an IMS database manager fails.
RM	SNGLRM	RM is defined without a resource structure, so only a single RM is allowed in the IMSplex.
RM	MULTRM	RM is defined with a resource structure, so multiple RMs are allowed in the IMSplex.

Output fields

The following table shows the output fields for the QUERY IMSPLEX command. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 36. Output fields for QUERY IMSPLEX command

Short label	Long label	Keyword	Meaning
IMSMBR	Member	N/A	IMSplex member name. The IMSplex member name is always returned.
IMSPLX	IMSplex	N/A	IMSplex name. The IMSplex name is always returned.
MBR	MbrName	N/A	IMSplex member that built the output line. The OM identifier of the OM that built the output line.
CC	CC	N/A	Completion code for the line of output. The completion code is always returned.
STT	Status	STATUS	IMSplex member status. For more information, see Table 33 on page 86.
JOB	JobName	JOB	Job name of IMSplex member.
OS	OSName	OS	Name of OS image on which the IMSplex member is executing.
STYP	Subtype	SUBTYPE	Subtype of IMSplex member. See Table 35 on page 87 for an explanation of the possible subtypes.

Table 36. Output fields for QUERY IMSPLEX command (continued)

Short label	Long label	Keyword	Meaning
TYP	Type	TYPE	IMSplex member type.
VER	Version	VERSION	IMSplex member version.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSPLEX command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 37. Return and reason codes for the QUERY IMSPLEX command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSPLEX command completed successfully.
X'02000008'	X'00002048'	The QUERY IMSPLEX command has more than one filter value specified. Only one of the TYPE or STATUS filters can be specified.
X'02000008'	X'0000203C'	An invalid keyword parameter value was specified.
X'0200000C'	X'00003000'	The QUERY IMSPLEX command is successful for at least one resource name. The QUERY IMSPLEX command is not successful for one or more resource names. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY IMSPLEX command are listed in the QUERY IMSPLEX completion code table.
X'0200000C'	X'00003004'	No resources were found to be returned. The resource names specified might be invalid or there were no resources that match the filter specified.
X'0200000C'	X'00003008'	The QUERY IMSPLEX command was routed to multiple clients. None of the clients that processed the command returned a return code and reason code to the OM. At least one command client returned either command response data or a response message.
X'02000014'	X'00005020'	The QUERY IMSPLEX command processing terminated. OM was unable to obtain storage for a system AWE while processing the command.
X'02000014'	X'0000502C'	The QUERY IMSPLEX command processing terminated. OM was unable to obtain storage for the command output header.
X'02000014'	X'00005030'	The QUERY IMSPLEX command processing terminated. OM was unable to obtain storage for the command output response.
X'02000014'	X'00005040'	The QUERY IMSPLEX command processing terminated because of an SCI error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains completion codes can be returned on a QUERY IMSPLEX command.

Table 38. Completion codes for the QUERY IMSPLEX command

Completion code	Meaning
0	The QUERY IMSPLEX command completed successfully for the resource.
4	The IMSplex name is unknown to the client that is processing the request. The IMSplex name might have been typed in error or the IMSplex might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

The following are examples of the QUERY IMSPLEX command:

Example 1 for QUERY IMSPLEX command

TSO SPOC input:

```
QRY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)
```

TSO SPOC output:

```
Response for: QUERY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS)
IMSplex  MbrName    CC Member   JobName  Type  Subtype  Status
CSLPLEX1 OM1OM       0 IMS2     IMS2     IMS   DBDC     READY,ACTIVE
CSLPLEX1 OM1OM       0 CQS1CQS  CQSRE1   CQS            ACTIVE
CSLPLEX1 OM1OM       0 PRAQJOB4 PRAQJOB4 DBRC  001      READY,ACTIVE
CSLPLEX1 OM1OM       0 DBRACSAH DBRACSAH DBRC  001      READY,ACTIVE
CSLPLEX1 OM1OM       0 SYS3     IMS1     IMS   DBDC     READY,ACTIVE
CSLPLEX1 OM1OM       0 OM1OM    OM1      OM             READY,ACTIVE
CSLPLEX1 OM1OM       0 IMS3     IMS3     IMS   DBDC     READY,ACTIVE
CSLPLEX1 OM1OM       0 PRAQJOB2 PRAQJOB2 DBRC  001      READY,ACTIVE
CSLPLEX1 OM1OM       0 USRT011  USRT011  AOP            ACTIVE
CSLPLEX1 OM1OM       0 RM1RM    RM1      RM    MULTRM   READY,ACTIVE
CSLPLEX1 OM1OM       0 SCI1SC   SCI1     SCI            READY,ACTIVE
CSLPLEX1 OM1OM       0 ODBM1OD  ODBM1    ODBM           READY,ACTIVE
```

OM API input:

```
CMD(QRY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 15:05:18.859217</statime>
<stotime>2002.163 15:05:18.860443</stotime>
<staseq>B7C4A41E663D11C3</staseq>
<stoseq>B7C4A41E6689B9C3</stoseq>
<rqsttkn1>USRT011 10080518</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
```

```
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>IMSPLEX         </kwd>
<input>QUERY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left"  />
<hdr slbl="MBR"    llbl="MbrName" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left"  />
<hdr slbl="CC"     llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="IMSMBR" llbl="Member"  scope="LCL" sort="N" key="0" scroll="NO"  len="8"
dtype="CHAR" align="left"  />
<hdr slbl="JOB"    llbl="JobName" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left"  />
<hdr slbl="TYP"    llbl="Type"    scope="LCL" sort="N" key="0" scroll="YES" len="5"
dtype="CHAR" align="left"  />
<hdr slbl="STYP"   llbl="Subtype" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left"  />
<hdr slbl="STT"    llbl="Status"  scope="GBL" sort="N" key="0" scroll="YES" len="*"
dtype="CHAR" align="left"  />
</cmdrsphdr>
<cmdrspdata>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(IMS2)      CC(   0) JOB(IMS2)      TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(CQS1CQS)   CC(   0) JOB(CQSRE1)    TYP(CQS)
STYP(  ) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(SYS3)      CC(   0) JOB(IMS1)      TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(PRAQJOB4)  CC(   0) JOB(PRAQJOB4)  TYP(DBRC)
STYP(001) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(DBRACSAH)  CC(   0) JOB(DBRACSAH)  TYP(DBRC)
STYP(001) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(OM1OM)     CC(   0) JOB(OM1)       TYP(OM)
STYP(  ) STT(EADY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(IMS3)      CC(   0) JOB(IMS3)      TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(PRAQJOB2)  CC(   0) JOB(PRAQJOB2)  TYP(DBRC)
STYP(001) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(USRT011)   CC(   0) JOB(USRT011)   TYP(AOP)
STYP(  ) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(RM1RM)     CC(   0) JOB(RM1)       TYP(RM)
STYP(MULTRM) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(SCI1SC)    CC(   0) JOB(SCI1)      TYP(SCI)
STYP(  ) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(ODBM1OD)   CC(   0) JOB(ODBM1)     TYP(ODBM)
STYP(  ) STT(READY,ACTIVE)</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY IMSPLEX command displays the IMSplex members that compose IMSplex CSLPLEX1. This IMSplex contains three IMS systems (IMS1, IMS2, and IMS3), a TSO SPOC (USRT011), a CQS (CQSRE1), RM (RM1), and OM (OM1). OM1 is the command master that built the output.

Example 2 for QUERY IMSPLEX command

TSO SPOC input:

```
QRY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)
```

TSO SPOC output:

```
 Response for: QUERY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)
IMSplex  MbrName    CC Member   JobName  Type  Subtype  Status
CSLPLEX1 OM1OM       0 IMS2     IMS2     IMS   DBDC     READY,ACTIVE
CSLPLEX1 OM1OM       0 CQS1CQS  CQSRE1   CQS            ACTIVE
CSLPLEX1 OM1OM       0 SYS3     IMS1     IMS   DBDC     READY,ACTIVE
```

```
CSLPLEX1 OM1OM       0 OM1OM    OM1      OM            READY,ACTIVE
CSLPLEX1 OM1OM       0 IMS3     IMS3     IMS   DBDC    READY,ACTIVE
CSLPLEX1 OM1OM       0 USRT011  USRT011  AOP           ACTIVE
CSLPLEX1 OM1OM       0 RM1RM    RM1      RM    MULTRM  READY,ACTIVE
CSLPLEX1 OM1OM       0 SCI1SC   SCI1     SCI           READY,ACTIVE
CSLPLEX1 OM1OM       0 ODBM1OD  ODBM1    ODBM          READY,ACTIVE
```

OM API input:

```
CMD (QRY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname><omvsn>1.1.0</omvsn>
<xmlvsn>1</xmlvsn>
<statime>2002.163 15:05:18.859217</statime>
<stotime>2002.163 15:05:18.860443</stotime>
<staseq>B7C4A41E663D11C3</staseq>
<stoseq>B7C4A41E6689B9C3</stoseq>
<rqsttkn1>USRT011 10080518</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>IMSPLEX         </kwd>
<input>QUERY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="MBR"    llbl="MbrName" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"     llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="IMSMBR" llbl="Member"  scope="LCL" sort="N" key="0" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="JOB"    llbl="JobName" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="TYP"    llbl="Type"    scope="LCL" sort="N" key="0" scroll="YES" len="5"
dtype="CHAR" align="left" />
<hdr slbl="STYP"   llbl="Subtype" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="STT"    llbl="Status"  scope="GBL" sort="N" key="0" scroll="YES" len="*"
dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(IMS2)    CC(   0) JOB(IMS2)   TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(CQS1CQS) CC(   0) JOB(CQSRE1) TYP(CQS)
STYP(   ) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(SYS3)    CC(   0) JOB(IMS1)   TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(OM1OM)   CC(   0) JOB(OM1)    TYP(OM)
STYP(   ) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(IMS3)    CC(   0) JOB(IMS3)   TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(USRT011) CC(   0) JOB(USRT011) TYP(AOP)
STYP(    ) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(RM1RM)   CC(   0) JOB(RM1)    TYP(RM)
STYP(MULTRM) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(SCI1SC)  CC(   0) JOB(SCI1)   TYP(SCI)
STYP(    ) STT(READY,ACTIVE)</rsp>
```

```
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(ODBM1OD) CC(   0) JOB(ODBM1)  TYP(ODBM)
STYP(    ) STT(READY,ACTIVE)</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY IMSPLEX command displays the IMSplex members that compose IMSplex CSLPLEX1. This IMSplex contains three IMS systems (IMS1, IMS2, and IMS3), a TSO SPOC (USRT011), a CQS (CQSRE1), RM (RM1), and OM (OM1). OM1 is the command master that built the output.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY LE command

Use the QUERY LE command to display Language Environment® (LE) runtime parameter overrides defined by a previous UPDATE LE command. The query can use filters on transaction code, LTERM, user ID, or program name.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 94
- "Usage notes" on page 95
- "Output fields" on page 95
- "Return, reason, and completion codes" on page 95
- "Examples" on page 96

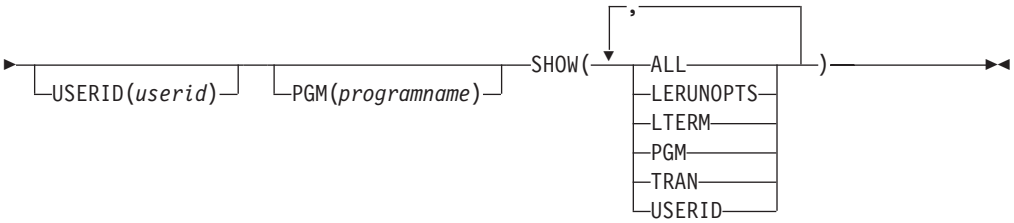
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY LE command and keywords can be issued.

Table 39. Valid environments for the QUERY LE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY LE	X	X	X
LTERM	X	X	X
PGM	X	X	X
SHOW	X	X	X
TRAN	X	X	X
USERID	X	X	X

Syntax



Keywords

The following keywords are valid for the QUERY LE command:

The parameters support a generic or wildcard parameter. A generic parameter is a 1-8 character name that includes an asterisk or a percent sign. An asterisk can be replaced by a zero or more characters to create a valid resource name. A percent sign can be replaced by exactly one character to create a valid resource name.

LTERM()
Specifies the 1-8 character name of the LTERM or LTERMs matching the generic or wildcard parameter.

PGM()
Specifies the 1-8 character name of the program or programs matching the generic or wildcard parameter.

SHOW()
Specifies the output fields to be returned. At least one SHOW field is required on the command.

ALL
Returns all the output fields. This is the same as if the following was specified: `SHOW(TRAN,LTERM,USERID,PGM,LERUNOPTS)`.

LERUNOPTS
Returns all of the LE override parameters associated with the transaction, LTERM, user ID, or program name.

LTERM
Returns the logical terminal name field.

PGM
Returns the program name field.

TRAN
Returns the transaction name field.

USERID
Returns the user identifier field.

TRAN()
Specifies the 1-8 character name of the transaction or transactions matching the generic or wildcard parameter. If the TRAN, LTERM, USERID, or PGM resource filters are not specified, all parameter overrides are returned.

USERID()
Specifies the 1-8 character name of the userid or userids matching the generic or wildcard parameter. If the TRAN, LTERM, USERID, or PGM resource filters are not specified, all parameter overrides are returned.

Usage notes

Any combination of parameters can be used to qualify the application instance. All entries found that match the criteria are returned. Specify on the command which output fields should be returned in the command response. You can ask for all information that includes transaction code, LTERM name, user ID, program name, and runtime parameters.

This command can be specified only through the Operations Manager API.

The command syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

The following table shows the QUERY LE output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 40. Output fields for QUERY LE command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code for the line of output. Completion code is always returned.
LTRM	Lterm	LTERM	LTERM Name requested by the QUERY.
MBR	MbrName	N/A	IMSplex member (IMS identifier) that built the output line. Member name is always returned.
PGM	Program	PGM	Program Name requested by the QUERY.
PRM	LERunOpts	LERUNOPTS	The LE override parameters for the specified resource filters.
TRAN	Trancode	TRAN	Transaction Name requested by the QUERY.
UID	Userid	USERID	Userid requested by the QUERY.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY LE command. The OM return and reason codes that may be returned as a result of the QUERY LE command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 41. Return and reason codes for the QUERY LE command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY LE command completed successfully.
X'00000008'	X'0000200C'	No resources found to return. Either the entry was previously deleted or a keyword filter was typed incorrectly.
X'00000008'	X'00002014'	An invalid character was specified in the filter name.
X'00000010'	X'00004040'	The parameter override header has not been initialized. Retry the command after restart is complete.
X'00000014'	X'00005000'	Unable to get storage from IMODULE GETSTOR.
X'00000014'	X'00005010'	Unable to obtain latch.
X'00000014'	X'00005FFF'	Internal IMS Error - Should not occur.

The following table includes an explanation of the completion code.

Table 42. Completion code for the QUERY LE command

Completion code	Meaning
0	The QUERY LE command completed successfully for the specified resource.

Examples

The following are examples of the QUERY LE command:

Example 1 for QUERY LE command

Assume the following filters are specified on QRY LE commands:

1. TRAN(PART) SHOW(ALL) Returns entries #1, 2, 3, 5, 6, 8.
2. TRAN(PART) LTERM(TERM1) SHOW(ALL) Returns entries #3, 5, 6.
3. LTERM(TERM2) USERID(BETTY) SHOW(ALL) Returns entry #7.
4. TRAN(PART) LTERM(TERM1) USERID(BETTY) SHOW(ALL) Does not return any entries.
5. TRAN(PART) LTERM(TERM*) SHOW(ALL) Returns entries #3, 5, 6, 8.
6. USERID(B*) SHOW(ALL) Returns entries #2, 5, 6, 7.

Rules for matching an entry which results in it being returned on QUERY command:

- If a filter is specified on the command for a particular resource it must match the resource filter defined in the entry. The resource in the QUERY LE command may be specified with wildcards as defined previously.
- A resource filter that is not specified on a QUERY LE command will match on any filter for the specific resource defined in the entry. A non-specified filter is treated as a wildcard. For instance if the LTERM filter is not specified on a QRY LE command it will match on any LTERM resource defined in an entry, as if LTERM(*) was specified on the command.

The following table is a logical representation of the parameter override table entries prior to any of the above query commands being processed.

Table 43. Parameter override table entries for example 1

Entry#	TRAN	LTERM	USERID	PROGRAM	LERUNOPTS
1	PART			DFSSAM02	aaaa
2	PART		BETTY		bbbb
3	PART	TERM1			cccc
4				DFSSAM02	dddd
5	PART	TERM1	BARBARA		eeee
6	PART	TERM1	BOB		ffff
7		TERM2	BETTY		gggg
8	PART	TERM2			iiii

Example 2 for QUERY LE command

TSO SPOC input:

```
QRY LE SHOW(ALL)
```

TSO SPOC output:

```
SYS3 0                        IAPMDI29 CCCC
SYS3 0 IAPMDI26      USRT001      RPTOPTS=((ON),NOOVR),RPTSTG=(((OFF),NOOVR)
SYS3 0 IAPMDI27 IMS1 USRT001  IAPMDI27 AAAA
```

OM API input:

```
CMD(QRY LE SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 17:34:01.196902</statime>
<stotime>2002.163 17:34:01.197368</stotime>
<staseq>B7C4C55B67566505</staseq>
<stoseq>B7C4C55B67738365</stoseq>
<rqsttkn1>USRT002 10103401</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>LE              </kwd>
<input>QRY LE SHOW(ALL)          </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"  llbl="MbrName"   scope="LCL" sort="A" key="1" scroll="NO" len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"   llbl="CC"        scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT" align="right" />
<hdr slbl="TRAN" llbl="Trancode"  scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="LTRM" llbl="Lterm"     scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
<hdr slbl="UID"  llbl="Userid"    scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
<hdr slbl="PGM"  llbl="Program"   scope="LCL" sort="N" key="0" scroll="YES" len="8"
```

```
dtype="CHAR" align="left" />
<hdr slbl="PRM"  llbl="LERunOpts" scope="LCL" sort="N" key="0" scroll="YES" len="*"
dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3    ) CC(   0) TRAN(        ) LTRM(      ) UID(        ) PGM(IAPMDI29)
PRM(CCCC    ) </rsp>
<rsp>MBR(SYS3    ) CC(   0) TRAN(IAPMDI26) LTRM(      ) UID(USRT001 ) PGM(        )
PRM(RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR)       )</rsp>
<rsp>MBR(SYS3    ) CC(   0) TRAN(IAPMDI27) LTRM(IMS1  ) UID(USRT001 ) PGM(IAPMDI27)
PRM(AAAA    ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The SHOW(ALL) parameter is specified, so all four filters and the runtime option string are shown for each table entry. Furthermore, no filters are specified in the command, so all table entries are shown. In this example, there are three table entries. The first specifies one filter (program) and the parameter string for this entry is CCCC. The second entry specifies two filters, trancode and user ID, and its parameter string is RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR). The last entry specifies all four filters and a parameter string of AAAA.

Example 3 for QUERY LE command

TSO SPOC input:

```
QRY LE SHOW(LTERM,USERID)
```

TSO SPOC output:

```
MbrName    CC Lterm    Userid
SYS3        0
SYS3        0          USRT001
SYS3        0 IMS1     USRT001
```

OM API input:

```
CMD(QRY LE SHOW(LTERM,USERID))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 17:36:27.588393</statime>
<stotime>2002.163 17:36:27.589261</stotime>
<staseq>B7C4C5E703729D6F</staseq>
<stoseq>B7C4C5E703A8D467</stoseq>
<rqsttkn1>USRT002 10103627</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>LE              </kwd>
<input>QRY LE SHOW(LTERM,USERID)          </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"  llbl="MbrName" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"   llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
```

```
dtype="INT" align="right" />
<hdr slbl="LTRM" llbl="Lterm"   scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
<hdr slbl="UID"  llbl="Userid"  scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3    ) CC(   0) LTRM(        ) UID(        ) </rsp>
<rsp>MBR(SYS3    ) CC(   0) LTRM(        ) UID(USRT001 ) </rsp>
<rsp>MBR(SYS3    ) CC(   0) LTRM(IMS1    ) UID(USRT001 ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: This command uses the SHOW parameter to limit the amount of data that is shown for each entry in the table. All three table entries are shown, but only the LTERM and TRAN filters are shown for each one. The first entry has neither an LTERM filter nor a USERID filter defined, so it is blank except for the MbrName and CC.

Example 4 for QUERY LE command

TSO SPOC input:

```
QRY LE USERID(USRT*) SHOW(LTERM,USERID)
```

TSO SPOC output:

```
MbrName    CC Lterm    Userid
SYS3        0          USRT001
SYS3        0 IMS1     USRT001
```

OM API input:

```
CMD(QRY LE USERID(USRT*) SHOW(LTERM,USERID))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 17:50:24.925819</statime>
<stotime>2002.163 17:50:24.926381</stotime>
<staseq>B7C4C9058F87B484</staseq>
<stoseq>B7C4C9058FAAD324</stoseq>
<rqsttkn1>USRT002 10105024</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>LE   </kwd>
<input>QRY LE USERID(USRT*) SHOW(LTERM,USERID)  </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"  llbl="MbrName" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"   llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT" align="right" />
<hdr slbl="LTRM" llbl="Lterm"   scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
<hdr slbl="UID"  llbl="Userid"  scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
```

```
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3    ) CC(   0) LTRM(        ) UID(USRT001 ) </rsp>
<rsp>MBR(SYS3    ) CC(   0) LTRM(IMS1    ) UID(USRT001 ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: This command uses both the USERID filter and the SHOW parameter to limit the amount of data shown in the output. The USERID(USRT*) parameter limits the output to only those table entries that define a USERID filter that fits the specified pattern (USRT*). The SHOW parameter limits how much information is returned about each table entry In this case, only the LTERM and USERID are shown.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

"UPDATE LE command" on page 701

QUERY LTERM command

Use the QUERY LTERM command to display information about logical terminals (LTERMs). This command can be specified only through the OM API and is valid on an XRF alternate.

Subsections:

- "Environment"
- "Syntax" on page 101
- "Keywords" on page 102
- "Usage notes" on page 107
- "Equivalent IMS type-1 commands" on page 108
- "Output fields" on page 108
- "QUERY LTERM status" on page 111
- "Return, reason, and completion codes" on page 112
- "Examples" on page 114

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

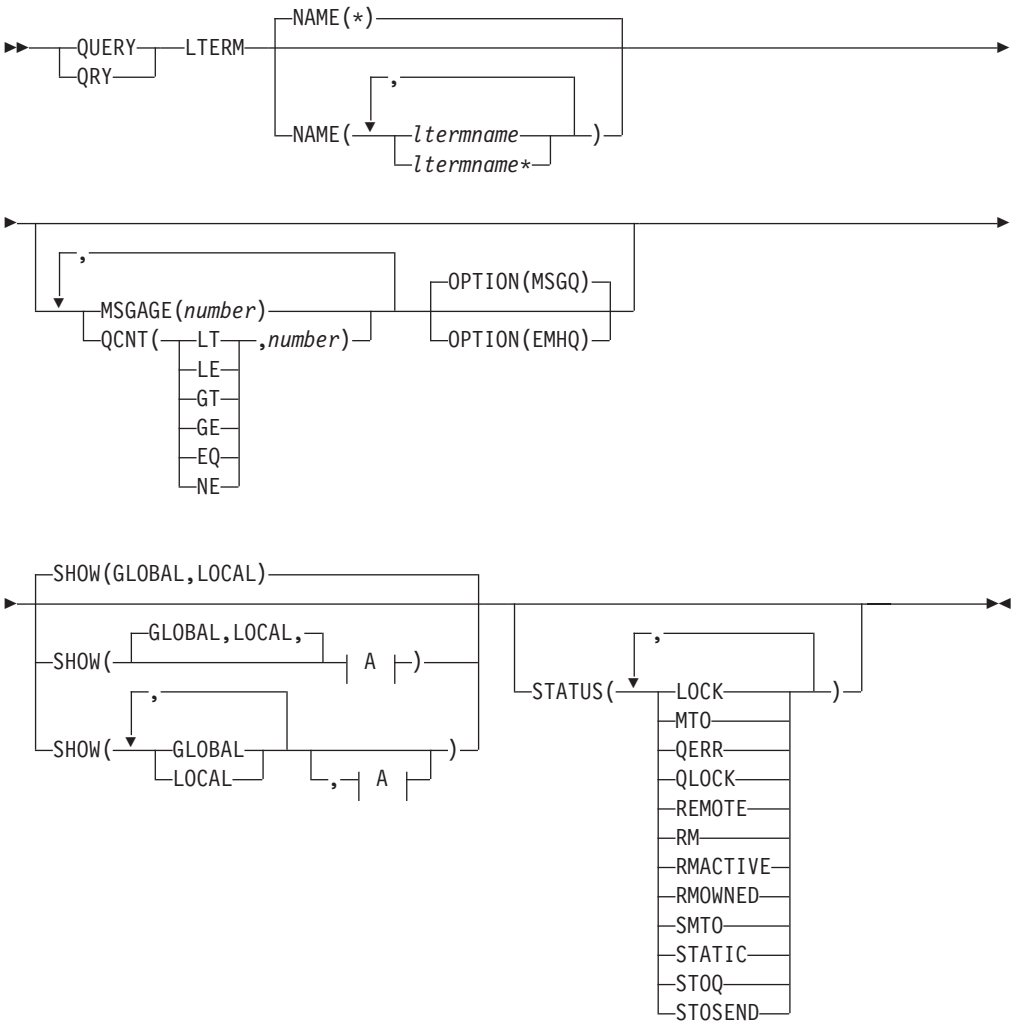
Table 44. Valid environments for the QUERY LTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY LTERM	X		X
NAME	X		X
MSGAGE	X		X
QCNT	X		X
OPTION	X		X
SHOW	X		X

Table 44. Valid environments for the QUERY LTERM command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
STATUS	X		X

Syntax



A:

Keywords

The following keywords are valid for the QUERY LTERM command:

MSGAGE()
Selects only those logical terminals that have at least one message whose message age is greater than the message age value specified. This applies only to messages in the shared message queues. The value specified is the number of days. Valid values are 0-365.

The information displayed includes the total count of messages on the queue, the count of messages with a message age greater than or equal to the message age specified, the time stamp of the oldest message, and the time stamp of the newest message.

When the MSGAGE filter is specified, the output returned includes the queue count of the LTERM even if SHOW(QCNT) is not specified, and the message age information even if SHOW(MSGAGE) is not specified.

MSGAGE( ) applies only to a shared-queues environment, and is processed only by the command master IMS. It is valid only when SHOW(GLOBAL) is specified.

The OPTION keyword tells IMS whether the message queue structure (OPTION(MSGQ)) or the EMH queue structure (OPTION(EMHQ)) is to be read. OPTION(MSGQ) is the default. OPTION(EMHQ) is valid only if shared EMH is used.

The performance implication is that in order to obtain message age information, all of the shared-queues LTERM messages on the Coupling Facility must be read.

If MSGAGE( ) is specified with the QCNT( ) or STATUS( ) filters, then IMS selects those LTERMs that match any of the specified filters (they do not need to match all specified filters).

NAME()
Specifies the names of one or more logical terminals that are to be displayed. Valid names are 1-8 characters, and wildcards can be specified. To display all logical terminals, specify NAME(*). NAME(*) is the default.

There are some special considerations when specifying certain specific LTERM names.

The primary and secondary MTO LTERM names may have generic names associated with them, as specified by the PMTOG and SMTOG keywords in the DFSDCxxx PROCLIB member. To display the primary or secondary MTO LTERMs on all systems, the operator should specify the generic LTERM name. When this is done, the output displayed will show the real LTERM names on each system, not the generic name.

The system console LTERM has a generic name of 'WTOR'. When shared queues are not enabled, 'WTOR' is also the real name of the system console LTERM. When shared queues are enabled in a non-XRF system, the real name is the IMS system ID (IMSID). When shared queues are enabled in an XRF system, the real name is the Recoverable Service Element (RSENAME). To display the system console LTERMs on all systems, the operator should specify the generic LTERM name 'WTOR'. When this is done, the output displayed will show the real LTERM names on each system, not the generic name.

Logical terminals associated with VTAM nodes might exist either locally, in the resource structure, or both. Logical terminals associated with the system console, SPOOL or SYSOUT devices will exist only in local systems.

OPTION()
Specifies additional information to process the command. This keyword is valid only with the MSGAGE() or QCNT() filter.

MSGQ
Specifies that queue count information should be retrieved from the shared message queue structure, or the local message queues if shared queues are not enabled (for QCNT filter only).

EMHQ
Specifies that queue count information should be retrieved from the expedited message handler queue structure. EMHQ is valid only when EMH queues are used.

QCNT()
Selects only those logical terminals that have a queue count less than (LT), less than or equal to (LE), greater than (GT), greater than or equal to (GE), equal to (EQ), or not equal to (NE) the specified number. The specified number cannot be a 1 when LT is specified, and cannot be a 0 when EQ, GE, LE, or LT is specified. Regardless of the condition specified, only LTERMs with a queue count greater than 0 are returned when the QCNT filter is specified. Lterms with a queue count of 0 are not returned.

The QCNT filter is valid in both a shared-queues environment and in a non-shared-queues environment.

In a shared-queues environment, the global queue count values are used to determine the LTERMs to be displayed. Therefore, SHOW(LOCAL) is invalid with the QCNT filter. In this environment, only the command master processes the command. All other IMS systems ignore the command. The OPTION keyword tells IMS whether the message queue structure (OPTION(MSGQ)) or the EMH queue structure (OPTION(EMHQ)) is to be read. OPTION(MSGQ) is the default. OPTION(EMHQ) is only valid if shared EMH is used. If QCNT is specified with a wildcard LTERM name, the performance implication is that all of the shared-queues LTERM messages on the Coupling Facility must be read.

In a non-shared-queues environment, the local queue count values are used to determine the LTERMs to be displayed. In this environment, the command is processed by each IMS that the command is routed to because the queues are local. Each IMS returns all of the LTERMs that it finds locally that match the

queue count filter specified. Because there is no local EMH queue count, OPTION(EMHQ) is invalid in this environment, and if specified, the command will be rejected.

When the QCNT filter is specified, the output returned includes the queue count of the LTERM, even if SHOW(QCNT) is not specified.

If QCNT() is specified with the MSGAGE() or STATUS() filters, then IMS selects those LTERMs that match any of the specified filters (they do not need to match all specified filters).

SHOW()

This specifies the LTERM output fields to be returned. If SHOW is not specified, and if none of the three keywords MSGAGE, QCNT, or STATUS is specified, then only the LTERM names are returned. This provides a method for a system management application to obtain a list of all LTERM names that are currently known in the IMSplex.

Two parameters, GLOBAL and LOCAL, are used to specify the location (global resources or local resources) where IMS should obtain the information that is to be displayed. The default is both GLOBAL and LOCAL.

The rest of the parameters are used to specify what information is displayed.

The parameters supported with the SHOW keyword, which can be specified in any order, are:

ALL
Returns all of the output fields.

COMPONENT
Displays the input and output component for the LTERM. The component numbers are defined in the NAME macro, user descriptor, signon user exit (DFSSGNX0), or the /ASSIGN LTERM command, and represent the component of a terminal that is used for input or output.

EMHQ
Displays the LTERM message queue count in the Expedited Message Handler (EMH) queues.

EMHQ is valid only when the GLOBAL parameter is specified on the SHOW keyword. If GLOBAL is not specified, then the EMHQ parameter is ignored.

EMHQ is processed only by the command master. It is ignored by all other IMS systems.

EMHQ is valid only if shared EMH is used in a shared queues environment. Otherwise, this parameter is ignored.

GLOBAL
The command master displays global information, depending on the other SHOW parameters specified. This includes information from shared queues and the resource structure.

The GLOBAL parameter is processed by the command master only. All other IMS systems ignore this parameter. If LOCAL is not also specified, then all IMS systems other than the command master ignore the command.

GLOBAL is applicable only when the command master is using shared queues or sysplex terminal management (or both).

GLOBAL is not applicable when the command master is not using shared queues or sysplex terminal management. In this environment, if LOCAL is also specified, then GLOBAL is ignored. Otherwise, the command master rejects the command.

If shared queues are enabled, and global queue counts are requested, then the command master will make requests to CQS to determine the appropriate queue counts. This includes both MSGQ and EMHQ.

If sysplex terminal management is enabled, then the command master will make requests to RM to determine the appropriate global status.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

LOCAL
All IMS systems including the command master display local information, depending on the other SHOW parameters specified. This includes information local to the IMS processing the command.

The LOCAL parameter is processed by all IMS systems, including the command master.

LOCAL is applicable in any environment, regardless of whether shared queues or sysplex terminal management are used.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

MSGAGE
Displays the count of messages whose ages are greater than the value specified by the MSGAGE() filter. If the MSGAGE() filter is not specified, then the value is assumed to be 0, and the count displayed is the total message queue count for the LTERM. The time stamp of the oldest and newest message on the LTERM queue is also displayed.

SHOW(QCNT) and SHOW(EMHQ) determine whether IMS reads the shared message queues, or the EMH queues. If neither is specified, then SHOW(QCNT) is assumed.

The performance implication is that in order to obtain message age information, all of the shared-queues LTERM messages on the Coupling Facility must be read.

The MSGAGE parameter is valid only in a shared-queues environment, and is ignored in all other environments. It is processed by the command master only. All other IMS systems ignore this parameter.

MSNAME
Displays the MSC logical link path name (msname) associated with the remote LTERM, and the remote and local system identifiers (SYSIDs). This applies only to LTERMs that are defined as MSC remote LTERMs.

NODE
Displays the node, if any, associated with the LTERM. For VTAM terminals, this is the terminal name. For LTERMs associated with

non-VTAM devices (system console, SPOOL, SYSOUT, and TCO), IMS displays a node name of DFSLN*xxx*, where *xxx* is the line number, and the line and PTERM number of the non-VTAM device.

OWNER
Displays the owner of the associated user or node resource in the resource structure. This applies only when sysplex terminal management is enabled, and is only processed by the command master. All other IMS systems ignore this parameter.

The owner is the IMSID (or RSENAME for XRF systems) of the IMS system that owns the associated user or node. An IMS system owns a user or node resource if the resource is active (the user is signed on, or the node is logged on), or an IMS system is maintaining significant status for that resource.

QCNT
Displays the LTERM message queue count.

The local queue counts value returned on this command represents the messages being processed by the IMS system where this command is issued. In a shared-queues environment, do not use the local queue counts for queue evaluation purposes. The global queue counts, which are returned on this command, represent the current message queue counts on the shared queues structure.

If the LOCAL parameter is also specified on the SHOW keyword, then all IMS systems that process the command, including the command master, display the local queue count. This is valid whether or not shared message queues are enabled.

If the GLOBAL parameter is also specified on the SHOW keyword, and shared message queues are enabled, then the command master displays the global queue count on the shared message queues (MSGQ).

The local and global queue counts are displayed as separate output fields.

STATUS
Returns local and global status of the LTERM. See "QUERY LTERM status" on page 111 for a list and meaning of possible status that can be returned.

USER
Displays the dynamic or ISC user, if any, associated with the LTERM.

VERSION
Displays the RM version number of the LTERM resource. This is the version number assigned to the LTERM, which is assigned by MVS, and maintained by RM, when the resource is created or updated in the resource structure. VERSION applies only when sysplex terminal management is enabled. VERSION is ignored when sysplex terminal management is not enabled.

STATUS()
Selects LTERMs for display that possess at least one of the specified LTERM statuses. The status might exist locally or globally if sysplex terminal management (STM) is enabled.

The STATUS filter is valid in both a sysplex terminal management environment and in a non sysplex terminal management environment.

In a sysplex terminal management environment, the status selected might exist locally, globally, or both. If sysplex terminal management is not enabled, then the status only exists locally.

If SHOW(LOCAL) is specified, then IMS will select only those LTERMs with the appropriate status in the local system. The command is processed by all IMS systems, including the command master.

If SHOW(GLOBAL) is specified, and sysplex terminal management is enabled, then IMS will select only those LTERMs with the appropriate status in the resource structure. The command is processed only by the command master.

If SHOW(GLOBAL) is specified, but sysplex terminal management is not enabled, then the command is rejected.

If SHOW(GLOBAL,LOCAL) is specified, which is the default, then IMS will select those LTERMs with the appropriate status either locally or in the resource structure (if sysplex terminal management is enabled). The command is processed by all IMS systems. The command master processes both global and local information.

The output returned when the status filter is specified includes the status of the LTERM, even if SHOW(STATUS) is not specified.

If STATUS() is specified with the MSGAGE() or QCNT() filters, then IMS selects those LTERMs that match any of the specified filters (they do not need to match all specified filters).

See "QUERY LTERM status" on page 111 to determine which filters can be used to select nodes with corresponding status.

Usage notes

The QUERY LTERM command can be specified only through the OM API.

QUERY LTERM can be issued on an XRF alternate system, but SHOW(GLOBAL) is not supported. Only local information can be displayed.

The processing of the QUERY LTERM command is different depending on whether IMS sysplex terminal management is enabled.

- If IMS sysplex terminal management is not enabled, processing is local for each system. The results of type-1 and type-2 commands are similar.
- If IMS sysplex terminal management is enabled, type-1 and type-2 command processing is similar when displaying local information. However, they differ in how global information is displayed.
- For type-1 /DISPLAY commands with IMS sysplex terminal management enabled, the command master displays information from either the resource structure or the local system, but not both. If the resource being displayed is not owned by any system or is owned by the command master, the command master displays the global resource. However, if the resource is owned by a system other than the command master, the command master displays only the local resource, and the owning system is responsible for displaying the global resource.
- For type-2 QUERY commands with IMS sysplex terminal management enabled, the command master is the only system that displays global resource information, regardless of whether the resource is owned. In addition, the command master displays local resource information. All other IMS systems that

process the command display local resource information only. This approach allows more flexibility in displaying all information in an IMSplex.

The SHOW keyword determines which IMS systems process the command, and what information is displayed.

- If SHOW(GLOBAL) is specified, then the command master displays global information, which can include the global queue count if shared queues are enabled, and status from the resource structure if sysplex terminal management is enabled (STM=YES defined in DFSDCxxx PROCLIB member). This is true whether or not the LTERM is active on any particular IMS system. All other IMS systems ignore the GLOBAL parameter with return code X'00000004' and reason code X'00001000'.
- If SHOW(LOCAL) is specified, then each IMS system to which OM routes the command (including the command master) processes the command, and displays information that is local to each system.
- If both GLOBAL and LOCAL are specified, which is the default, then both global and local information are displayed. Each IMS system to which OM routes the command, including the command master, processes the command, and displays local information. In addition to local information, the command master displays global information.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY LTERM command and the type-1 IMS commands that perform similar functions.

Table 45. Type-1 equivalents for the QUERY LTERM command

QUERY LTERM command	Similar IMS type-1 command
QUERY LTERM SHOW(COMPONENT)	/DISPLAY ASMT LTERM ltermname
QUERY LTERM SHOW(EMHQ)	/DISPLAY LTERM ltermname QCNT EMHQ
QUERY LTERM SHOW(MSNAME)	/DISPLAY LTERM ltermname
QUERY LTERM SHOW(NODE)	/DISPLAY ASMT LTERM ltermname
QUERY LTERM SHOW(QCNT)	/DISPLAY LTERM ltermname /DISPLAY LTERM ltermname QCNT
QUERY LTERM SHOW(STATUS)	/DISPLAY LTERM ltermname /DISPLAY STATUS LTERM
QUERY LTERM SHOW(USER)	/DISPLAY ASMT LTERM ltermname
QUERY LTERM STATUS(MTO,SMTO)	/DISPLAY MASTER /RDISPLAY MASTER
QUERY LTERM MSGAGE(x)	/DISPLAY QCNT LTERM MSGAGE x
QUERY LTERM STATUS(status)	/DISPLAY STATUS LTERM

Output fields

The following table shows the QUERY LTERM output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

SHOW parameter
Identifies the parameter on the SHOW keyword that caused the field to be generated. *Error* appears for output fields that are returned for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field. GBL indicates that the field can be generated only by the command master when displaying global information for SHOW(GLOBAL). LCL indicates that the field can be generated by any IMS displaying local information for SHOW(LOCAL). N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 46. Output fields for the QUERY LTERM command

Short label	Long label	SHOW parameter	Scope	Meaning
CC	CC	N/A	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 112 for more information. The completion code is always returned.
CCTXT	CCText	*Error*	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
EAGE	EmhqAged	MSGAGE	GBL	Count of EMHQ messages with a message age greater than or equal to the message age specified by the MSGAGE() filter. If no filter is specified, the count is the total number of messages queued.
EMHQ	EMHQCnt	EMHQ	GBL	Global LTERM queue count in the EMH (Expedited Message Handler) queues. EMHQ is displayed only if shared EMH is used.
ETNEW	EmhqTStmpNew	MSGAGE	GBL	The time stamp of the newest EMHQ message for the LTERM on the shared queues.
ETOLD	EmhqTStmpOld	MSGAGE	GBL	The time stamp of the oldest EMHQ message for the LTERM on the shared queues.
GBL	Gbl	GLOBAL	GBL	If 'Y', then the output reflects the status found globally in shared queues or RM. If blank, then the output reflects the status found locally.
ICMP	InCmp	COMPONENT	GBL	The input component that the LTERM is assigned to in the resource structure.
LICMP	LInCmp	COMPONENT	LCL	The input component that the LTERM is assigned to in the local system.
LINE	Line	NODE	LCL	Identifies the line number for system console, SPOOL, SYSOUT, or TCO device.
LMSN	LMSName	MSNAME	LCL	The associated logical link path name when the LTERM is an MSC remote LTERM.

Table 46. Output fields for the QUERY LTERM command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LNODE	LNode	NODE	LCL	Identifies the dynamic or static node associated with the LTERM on the local system.
LOCMP	LOutCmp	COMPONENT	LCL	The output component that the LTERM is assigned to in the local system.
LQ	LQCnt	QCNT	LCL	Local queue count.
LSIDL	LSIDL	MSNAME	LCL	Local system identification of the associated logical link path name when the LTERM is an MSC remote LTERM.
LSIDR	LSIDR	MSNAME	LCL	Remote system identification of the associated logical link path name when the LTERM is an MSC remote LTERM.
LSTT	LclStat	STATUS	LCL	Local logical terminal status. See Table 47 on page 111 for information about the logical terminal status that can be returned.
LTERM	Lterm	N/A	N/A	Logical terminal name. The logical terminal name is always returned.
LUSER	LUser	USER	LCL	Identifies the dynamic or ISC user associated with the LTERM on the local system.
LVER	LVersion#	VERSION	LCL	Version number for the LTERM resource being maintained in the local system. This field applies only when STM is enabled.
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. The IMS identifier is always returned.
NODE	Node	NODE	GBL	Identifies the dynamic or static node associated with the LTERM in the resource structure.
OCMP	OutCmp	COMPONENT	GBL	The output component that the LTERM is assigned to in the resource structure.
OWNER	Owner	OWNER	GBL	The IMSID (or RSENAME for XRF systems) of the IMS system that “owns” the associated node or user resource. This field is returned only by the command master, and applies only when sysplex terminal management is enabled.
PTERM	PTerm	NODE	LCL	Identifies the PTERM number for system console, SPOOL, SYSOUT or TCO device.
QAGE	QCntAged	MSGAGE	GBL	Count of MSGQ messages with a message age greater than or equal to the message age specified by the MSGAGE() filter. If no filter is specified, the count is the total number of messages queued.
QCNT	QCnt	QCNT	GBL	Global queue count on the shared queues. Global queue count can be displayed only if shared queues are used.
QTNEW	TStmpNew	MSGAGE	GBL	The time stamp of the newest MSGQ message for the LTERM on the shared queues.
QTOLD	TStmpOld	MSGAGE	GBL	The time stamp of the oldest MSGQ message for the LTERM on the shared queues.

Table 46. Output fields for the QUERY LTERM command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
STT	Status	STATUS	GBL	Global logical terminal status from the resource structure. See Table 47 for information about the logical terminal status that can be returned.
USER	User	USER	GBL	Identifies the dynamic or ISC user associated with the LTERM in the resource structure.
VER	Version#	VERSION	GBL	Version number for the LTERM resource being maintained in the resource structure. This field applies only when STM is enabled.

QUERY LTERM status

The following table shows the possible LTERM status that can be displayed. The columns in the table are:

Status The LTERM status that is displayed.

STATUS parameter
The STATUS() filter that will select LTERMs with the specified status.

Scope The scope of the status. GBL indicates that the status can be global (it exists in the resource structure when STM is enabled), and is returned with the STT short label. LCL indicates that the status can be local, and is returned with the LSTT short label.

Meaning
Provides a brief description of the status.

Table 47. QUERY LTERM status

Status	STATUS parameter	Scope	Meaning
LOCK	LOCK	LCL	Logical terminal was locked by a /LOCK LTERM command.
MTO	MTO	LCL	Logical terminal is the master terminal for the local IMS.
QERR	QERR	LCL	I/O error has occurred on the queue for this logical terminal.
QLOCK	QLOCK	LCL	Logical terminal is locked from sending any further output or from receiving input that could create additional output for the same logical terminal until the state is reset by a specific request received on the session.
REMOTE	REMOTE	LCL	Logical terminal is defined remotely to this IMSplex, accessible via MSC.
RM	RM	GBL	The LTERM exists in the resource structure managed by RM.
RMACTIVE	RMACTIVE	GBL	The associated user or node is active (signed-on or logged-on) in the IMSplex, as indicated in the RM structure (RM active).
RMOWNED	RMOWNED	GBL	The associated user or node is owned by an IMS system in the IMSplex, as indicated in the RM structure (RM owned).

Table 47. QUERY LTERM status (continued)

Status	STATUS parameter	Scope	Meaning
SMTO	SMTO	LCL	The logical terminal is the secondary master terminal for the local IMS.
STATIC	STATIC	LCL and GBL	Logical terminal was defined during system definition.
STOQ	STOQ	LCL and GBL	Input is stopped for the logical terminal.
STOSEND	STOSEND	LCL and GBL	Output is stopped for the logical terminal.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY LTERM command. The OM return and reason codes that may be returned as a result of the QUERY LTERM command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 48. Return and reason codes for the QUERY LTERM command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001000'	The command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000008'	X'00002014'	An invalid character was specified in the resource name.
X'00000008'	X'00002040'	An invalid parameter value was specified. An invalid SHOW or STATUS value might have been specified. A value of 0 might have been specified for QCNT with LE, GE or, EQ. Or a value of 1 might have been specified for QCNT with LT.
X'0000000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See Table 49 on page 113 for details.
X'0000000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code. See Table 49 on page 113 for details.
X'00000010'	X'00004004'	Command processing terminated because CQS was not active.
X'00000010'	X'00004005'	Command processing terminated because CQS was not connected to the queue structure.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	Command processing terminated because the resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.

Table 48. Return and reason codes for the QUERY LTERM command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004104'	Command processing terminated because RM is not available.
X'00000010'	X'00004108'	Command processing terminated because SCI is not available.
X'00000014'	X'00005004'	A DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	Command processing terminated because of an RM error.
X'00000014'	X'00005104'	Command processing terminated because of a CQS error.
X'00000014'	X'00005108'	Command processing terminated because of an SCI error.
X'00000014'	X'00005FFF'	Command processing terminated because of an internal IMS error.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 49. Completion codes for the QUERY LTERM command

Completion code	Completion code text	Meaning
0		The QUERY LTERM command completed successfully for the resource.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
98	CQS REQUEST ERROR	Global queue counts could not be obtained because of a CQS error.
1A0	Lterm resource is in error	The LTERM resource was found in the resource structure. An associated resource was needed, but it was either not found or appeared to be in error. This is normally an error condition, but it could be a temporary condition caused by terminal or command activity. Retry the command.

Examples

The following are examples of the QUERY LTERM command:

Example 1 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(LTERM0*,XYZ) SHOW(LOCAL)
```

TSO SPOC output:

```
Lterm    MbrName CC CCText
LTERM01  IMS1     0
LTERM02  IMS1     0
LTERM02  IMS2     0
LTERM03  IMS2     0
LTERM04  IMS1     0
LTERM04  IMS2     0
LTERM05  IMS1     0
XYZ      IMS1    10 NO RESOURCES FOUND
XYZ      IMS2    10 NO RESOURCES FOUND
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM is irrelevant because only LOCAL information is requested. Shared queues are irrelevant because only LOCAL information is requested. IMS1, the command master, displays only local information because no global information is requested. IMS2 displays local information only.

- LTERM01 exists on IMS1 only.
- LTERM02 exists on IMS1 and IMS2.
- LTERM03 exists on IMS2 only.
- LTERM04 exists on IMS1 and IMS2.
- LTERM05 exists on IMS1 only.
- XYZ does not exist on any system.

Example 2 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(LTERM0*)
```

TSO SPOC output:

```
Lterm    MbrName CC  Gbl
LTERM01  IMS1     0
LTERM02  IMS1     0  Y
LTERM02  IMS1     0
LTERM02  IMS2     0
LTERM03  IMS2     0
LTERM04  IMS1     0  Y
LTERM04  IMS1     0
LTERM04  IMS2     0
LTERM05  IMS1     0  Y
LTERM05  IMS1     0
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information only.

- LTERM01 exists on IMS1 only.
- LTERM02 exists on IMS1 and IMS2, and in the resource structure.
- LTERM03 exists on IMS2 only.

- LTERM04 exists on IMS1, IMS2, and in the resource structure.
- LTERM05 exists on IMS1 and in the resource structure.

Example 3 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(LTERM0*) SHOW(GLOBAL,STATUS,QCNT,OWNER)
```

TSO SPOC output:

```
Lterm    MbrName CC  Gbl   QCnt  Owner   Status
LTERM02  IMS1     0  Y        0  IMS2    STATIC,RM,RMACTIVE,RMOWNED
LTERM03  IMS1     0  Y        1
LTERM04  IMS1     0  Y        3  IMS2    STATIC,RM,RMOWNED
LTERM05  IMS1     0  Y        0          STOQ,STOSEND,RM
LTERM06  IMS1     0  Y        2
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are enabled. IMS1, the command master, displays global information. IMS2 ignores the command (RC=4, RSN=x1000) because SHOW(GLOBAL) is specified.

- LTERM02 exists in the resource structure, is currently active on IMS2, and has no messages in the shared queues.
- LTERM03 does not exist in the resource structure, but has one message in the shared queues.
- LTERM04 exists in the resource structure, is not currently active, but is owned by IMS2 which indicates status exists on IMS2, and has three messages in the shared queues.
- LTERM05 exists in the resource structure, is stopped, is not currently active or owned, and has no messages in the shared queues.
- LTERM06 does not exist in the resource structure, but has two messages in the shared queues.

Example 4 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(LTERM01,LTERM02,LTERM04) SHOW(QCNT,EMHQ)
```

TSO SPOC output:

```
Lterm    MbrName CC CCText             Gbl   QCnt  EMHQCnt  LQCnt
LTERM01  IMS1     0                    Y        0        0
LTERM01  IMS1     0                                          0
LTERM01  IMS2    10 NO RESOURCES FOUND
LTERM02  IMS1     0                    Y        0        1
LTERM02  IMS1     0                                          0
LTERM02  IMS2     0                                          0
LTERM04  IMS1     0                    Y        3        0
LTERM04  IMS1     0                                          0
LTERM04  IMS2     0                                          0
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are enabled. IMS1, the command master, displays global and local information. IMS2 displays local information only.

- LTERM01 exists on IMS1 only, and has no messages in shared queues.
- LTERM02 exists on IMS1 and IMS2, and has one message in shared EMH.
- LTERM04 exists on IMS1 and IMS2, and has three messages in shared queues.

Example 5 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM MSGAGE(7) QCNT(GE,3)
```

TSO SPOC output:

```
Lterm    MbrName CC Gbl   QCnt  QCntAged TStmpOld     TStmpNew
LTERM04  IMS1    0  Y        3         0 07128/140758 07128/141005
LTERM06  IMS1    0  Y        2         2 07120/110113 07120/110240
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is irrelevant because global status other than queue counts is not requested. Shared queues are enabled. IMS1, the command master, displays global information. IMS2 ignores the command (RC=4, RSN=x1000) because QCNT() in a shared-queues environment is processed by the command master only.

- LTERM04 has three messages queued, which satisfies the QCNT filter, and all messages are less than 7 days old.
- LTERM06 has two messages queued that are older than 7 days, which satisfies the MSGAGE filter.

Example 6 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(*) QCNT(GE,2)
```

TSO SPOC output:

```
Lterm    MbrName CC  Gbl   QCnt
LTERM04  IMS1     0  Y        3
LTERM06  IMS1     0  Y        2
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM is irrelevant because only queue information is requested. Shared queues are enabled. IMS1, the command master, displays global information. IMS2 ignores the command (RC=4, RSN=x1000) because QCNT() in a shared-queues environment is processed by the command master only. The QCnt output is displayed even if SHOW(QCNT) was not specified.

- LTERM04 has three messages in the shared queues.
- LTERM06 has two messages in the shared queues.

Example 7 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(LTERM04,LTERM05,LTERM11) SHOW(LOCAL,ALL)
```

TSO SPOC output:

```
(screen 1)
Lterm    MbrName CC CCText             LQCnt  LNode   LInCmp LOutCmp
LTERM04  IMS1     0                        0  NODE04       1       1
LTERM04  IMS2     0                        0  NODE04       1       1
LTERM05  IMS1     0                        0               1       1
LTERM05  IMS2    10 NO RESOURCES FOUND
LTERM11  IMS1    10 NO RESOURCES FOUND
LTERM11  IMS2     0                        0

(scrolled right to screen 2)
Lterm    MbrName LUser   LMSName  LSIDR LSIDL LVersion# LclStat
LTERM04  IMS1                                         0 STATIC
LTERM04  IMS2                                         7 STATIC
LTERM05  IMS1    USER05                               2 STOQ,STOSEND
```

```
LTERM05  IMS2
LTERM11  IMS1
LTERM11  IMS2            LINK2111    21    11         0 STATIC,REMOTE
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM is enabled. Shared queues are irrelevant because global queue counts are not requested. IMS1, the command master, displays local information only because only LOCAL is specified. IMS2 displays local information only.

- LTERM04 exists on IMS1 and IMS2. The LTERM is statically allocated to NODE04 on all systems.
- LTERM05 exists on IMS1 only. The LTERM is dynamic, allocated to USER05, and is stopped. There is no node because the user is not signed on.
- LTERM11 exists on IMS2 only. It is defined as a remote LTERM. The associated logical link name is LINK2111, and the remote and local SIDs are 21 and 11.

Example 8 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(LTERM04,LTERM05,LTERM11) SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
Lterm    MbrName CC CCText             Gbl   QCnt  EMHQCnt  Owner  Node
LTERM04  IMS1     0                    Y        0        0  IMS2   NODE04
LTERM04  IMS1     0
LTERM04  IMS2     0
LTERM05  IMS1     0                    Y        0        0
LTERM05  IMS1     0
LTERM05  IMS2    10 NO RESOURCES FOUND
LTERM11  IMS1    10 NO RESOURCES FOUND Y
LTERM11  IMS1    10 NO RESOURCES FOUND
LTERM11  IMS2     0

(scrolled right to screen 2)
Lterm    MbrName Gbl  InCmp OutCmp User     Version#  QcntAged
LTERM04  IMS1    Y        1      1                7         3
LTERM04  IMS1
LTERM04  IMS2
LTERM05  IMS1    Y        1      1 USER05         2         0
LTERM05  IMS1
LTERM05  IMS2
LTERM11  IMS1    Y
LTERM11  IMS1
LTERM11  IMS2

(scrolled right to screen 3)
Lterm    MbrName Gbl  TStmpOld     TStmpNew     EmhqAged
LTERM04  IMS1    Y    07128/140758 07128/141005        0
LTERM04  IMS1
LTERM04  IMS2
LTERM05  IMS1    Y                                     0
LTERM05  IMS1
LTERM05  IMS2
LTERM11  IMS1    Y
LTERM11  IMS1
LTERM11  IMS2

(scrolled right to screen 4)
Lterm    MbrName Gbl  EmhqTStmpOld EmhqTStmpNew Status
LTERM04  IMS1    Y                             STATIC,RM,RMOWNED
LTERM04  IMS1
LTERM04  IMS2
LTERM05  IMS1    Y                             STOQ,STOSEND,RM
```

```
LTERM05  IMS1
LTERM05  IMS2
LTERM11  IMS1    Y
LTERM11  IMS1
LTERM11  IMS2

(scrolled right to screen 5)
Lterm    MbrName Gbl  LQCnt  LNode   LInCmp LOutCmp LUser   LMSName
LTERM04  IMS1    Y
LTERM04  IMS1             0  NODE04       1       1
LTERM04  IMS2             0  NODE04       1       1
LTERM05  IMS1    Y
LTERM05  IMS1             0               1       1 USER05
LTERM05  IMS2
LTERM11  IMS1    Y
LTERM11  IMS1
LTERM11  IMS2             0                                 LINK2111

(scrolled right to screen 6)
Lterm    MbrName Gbl  LSIDR LSIDL LVersion# LclStat
LTERM04  IMS1    Y
LTERM04  IMS1                             0 STATIC
LTERM04  IMS2                             7 STATIC
LTERM05  IMS1    Y
LTERM05  IMS1                             2 STOQ,STOSEND
LTERM05  IMS2
LTERM11  IMS1    Y
LTERM11  IMS1
LTERM11  IMS2            21    11         0 STATIC,REMOTE
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM and shared queues are enabled. IMS1, the command master, displays global and local information. IMS2 displays local information only.

- LTERM04 exists on IMS1, IMS2, and in the resource structure. Global status indicates it is not currently active, but is owned by IMS2 which indicates significant status exists for the node on IMS2. The LTERM is statically allocated to NODE04 on all systems.
- LTERM05 exists on IMS1 and in the resource structure. LTERM05 is dynamic. Global status indicates it is stopped, but is not active or owned on any system. The local status on IMS1 indicates it is stopped. The LTERM is allocated to user USER05 in both IMS1 and RM. There is no node because the user is not signed on. LTERM11 exists on IMS2 only, so there is no global status. It is defined as a remote LTERM. The associated logical link name is LINK2111, and the remote and local SIDs are 21 and 11.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY ASMT command (Commands)

/DISPLAY LTERM command (Commands)

/DISPLAY MASTER command (Commands)

/DISPLAY QCNT command (Commands)

/DISPLAY STATUS command (Commands)

Command keywords and their synonyms (Commands)

Chapter 10, "/RDISPLAY command," on page 389

QUERY MEMBER command

Use the QUERY MEMBER command to display status or attribute information about one or more members of the IMSplex.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 120
- "Output fields" on page 124
- "Return, reason, and completion codes" on page 125
- "Examples" on page 125

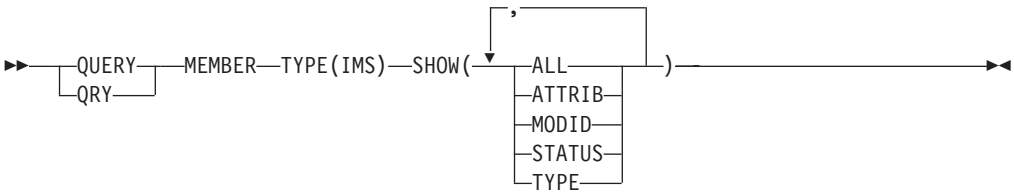
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY MEMBER command and keywords can be issued.

Table 50. Valid environments for the QUERY MEMBER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY MEMBER	X	X	X
ALL	X	X	X
ATTRIB	X	X	X
SHOW	X	X	X
STATUS	X	X	X
TYPE	X	X	X

Syntax



Keywords

The following keywords are valid for the QUERY MEMBER command:

SHOW()
Specifies the output fields to be returned.

ALL
Returns all the output fields.

ATTRIB
Displays the IMSplex member attributes. These are static definitions.

LOCAL
Displays the local status information about one or more members of the IMSplex.

GLOBAL
Displays the global status information about one or more members of the IMSplex.

MODID
Displays the online change modify id. The modify id is incremented by each successful online change. During a global online change, an IMS's local modify id indicates whether the IMS has reached the online change commit phase 2 and is synchronized with the OLCSTAT data set. If the modify id is initialized to zero by the Global Online Change utility, the modify id represents the number of global online changes that have successfully completed.

The local modify id on an XRF alternate system indicates whether the XRF alternate has gotten the X'70' log record, performed online change, and is synchronized with the OLCSTAT data set. If the XRF alternate's modid matches the OLCSTAT data set modid as displayed by the QUERY OLC LIBRARY (OLCSTAT) command, the XRF alternate is synchronized with the OLCSTAT data set.

STATUS
Displays the IMSplex member status. IMSplex member status can change dynamically.

TYPE
Shows the IMSplex member type. The IMSplex member type can be IMS, for the IMS address space.

TYPE()
Specifies the IMSplex member type for which information will be displayed. IMS is the only type available.

Usage notes

The QUERY MEMBER command can be specified only through the OM API.

The QUERY MEMBER command indicates whether dynamic resource definition for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=DYN) or not. When QUERY MEMBER SHOW(ALL) or SHOW(ATTRIB) is specified and DRD is enabled (MODBLKS=DYN), QUERY MEMBER output displays DYNMODBLKS as the attribute under the LclAttr column.

There are two commands named QUERY MEMBER. QUERY MEMBER TYPE(IMS) is a type-2 command that gets information about IMS members from TSO SPOC or the OM API. QUERY MEMBER TYPE(IMSCON) is a z/OS modify command that gets information about IMS Connect members. See IMS Connect QUERY MEMBER command (Commands) for more information.

QUERY MEMBER attributes

The following table shows the possible IMS attributes. The table contains information about attributes such as the attribute that is returned, the scope of the attribute, and the meaning of the attribute. Global attributes are returned with the ATTR short label. Local attributes are returned with the LATTR short label.

Table 51. Attributes for QUERY MEMBER command

Attribute	Scope	Meaning
DYNMODBLKS	LCL	Dynamic resource definition is enabled for MODBLKS resources
GBLOLC	LCL	Global online change is enabled.
NO-STM	LCL	IMS is not sharing terminal resources because STM=NO was specified in DFSDCxxx PROCLIB member or IMS is registered to an RM that is not using a resource structure.
RMENVNO	LCL	No RM environment is running.
RSRTRK	LCL	Remote Site Recovery tracker.
SHAREDQ	LCL	Shared queues are enabled.

QUERY MEMBER status

The following table shows the possible member status. The table contains information about status such as the status that is returned, the scope of the status, and the meaning of the status. Global status is returned with the STT short label. Local status is returned with the LSTT short label. A scope of LCL means that the status is local to the IMS specified and is returned with the LSTT short label. A scope of GBL means that the status is global to all the IMS systems and is returned with the STT short label.

Table 52. Status for QUERY MEMBER command

Status	Scope	Meaning
LEOPT	LCL	Language Environment options are enabled for this IMS.
OLCABRTC	LCL	Online change abort completed. A TERMINATE OLC command or /MODIFY ABORT command is entered. Online change abort phase completed locally for this IMS. The IMS is taken out of the online change state.
OLCABRTI	LCL	Online change abort in progress. A TERMINATE OLC command or /MODIFY ABORT command is entered. Online change abort phase is in progress locally for this IMS.
OLCCMT1C	LCL or GBL	Online change commit phase 1 completed. An INITIATE OLC PHASE(COMMIT) command or /MODIFY COMMIT command is entered. Online change commit phase 1 completed either locally for the IMS, or globally for all of the IMS systems in the IMSplex. After all of the IMS systems have attempted commit phase 1, the online change master updates the OLCSTAT data set and the online change is considered to be complete. Type-1 commands from the system console, an IMS terminal, or the MTO are queued while the IMS is in this state. Queued commands are processed after the online change is committed or aborted. If the type-1 command is entered from the system console, the WTOR does not appear until this IMS is out of the online change state.

Table 52. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
OLCCMT1I	LCL or GBL	Online change commit phase 1 in progress. An INITIATE OLC PHASE(COMMIT) command or /MODIFY COMMIT command is entered. Online change commit phase 1 is in progress either locally for this IMS or globally for all the IMS systems in the IMSplex.
OLCCMT2C	LCL or GBL	Online change commit phase 2 completed. An INITIATE OLC PHASE(COMMIT) command or a /MODIFY COMMIT command is entered. Online change commit phase completed either locally for this IMS or globally for all the IMS systems in the IMSplex. Type-1 commands from the system console, an IMS terminal, or the MTO are queued while the IMS is in this state. Queued commands are processed after the online change is committed or aborted. If the type-1 command is entered from the system console, the WTOR does not appear until this IMS is out of the online change state.
OLCCMT2F	LCL	Online change commit phase 2 failed. An INITIATE OLC PHASE(COMMIT) command or a /MODIFY COMMIT command is entered. Online change commit phase 2 failed locally for this IMS. This IMS may be stuck in an online change state, where the TERMINATE OLC command or /MODIFY ABORT command does not work. If that is the case, cancel the IMS and warm start IMS. This IMS can warm start, since it successfully participated in the online change except for commit phase 2.
OLCCMT2I	LCL or GBL	Online change commit phase 2 in progress. An INITIATE OLC PHASE(COMMIT) command or a /MODIFY COMMIT command is entered. Online change commit phase 2 is in progress either locally for this IMS or globally for all the IMS systems in the IMSplex.
OLCCMT3C	GBL	Online change commit phase 3 completed. An INITIATE OLC PHASE(COMMIT) command is entered. Online change commit phase 3 is completed globally on the other IMS systems except for the master. The COMMIT master still needs to perform commit phase 3 locally. The online change is committed, but commit phase 3 is still needed to clean up the online change information on all the IMS systems.
OLCCMT3F	GBL	Online change commit phase 3 failed. An INITIATE OLC PHASE(COMMIT) command is entered. Online change commit phase 3 failed globally on the other IMS systems. The master skips attempting to perform commit phase 3 locally and exits with an error, leaving itself in a global online change state. The other IMS systems may or may not have actually completed commit phase 3. Issue another COMMIT command to the previous COMMIT command master to complete the online change.

Table 52. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
OLCCMT3I	LCL or GBL	Online change commit phase 3 is in progress. An INITIATE OLC PHASE(COMMIT) command is entered. Online change commit phase 3 is in progress either locally for this IMS or globally for all the IMS systems in the IMSplex. The online change is committed, but commit phase 3 is needed to clean up online change information on all the IMS systems.
OLCMACB	GBL	Member online change for ACBLIB in progress.
OLCMSTR	GBL	Online change phase master. An INITIATE OLC PHASE(PREPARE), an INITIATE OLC PHASE(COMMIT), or a TERMINATE OLC command is entered. This IMS is the master of the online change phase currently in progress, either prepare, commit, or terminate. A different IMS may be master of each phase of online change.
OLCPREPC	LCL or GBL	Online change prepare phase completed. An INITIATE OLC PHASE(PREPARE) command or a /MODIFY PREPARE command is entered. Online change prepare phase completed locally for this IMS or globally for all the IMS systems in the IMSplex.
OLCPREPF	LCL	Online change prepare phase failed. An INITIATE OLC PHASE(PREPARE) command is entered. Online change prepare phase failed locally for this IMS. A TERMINATE OLC is required to delete the MWA created for the online change and also to delete the online change process that was initiated with RM.
OLCPREPI	LCL or GBL	Online change prepare phase in progress. An INITIATE OLC PHASE(PREPARE) command or a /MODIFY PREPARE command is entered. Online change prepare phase is in progress locally for this IMS or globally for all the IMS systems in the IMSplex.
OLCTERMC	GBL	Online change terminate completed. A TERMINATE OLC command was entered. Online change termination is completed for the IMS systems in the IMSplex.
OLCTERMF	LCL	TERMINATE FAILED: A TERMINATE OLC command is directed to the IMS that is not in an online change state. An MWA is created to coordinate the TERMINATE OLC command. The TERMINATE OLC command fails because of an RM, SCI, or CQS error and the MWA is set to a 'Terminate Failed' state. A subsequent TERMINATE OLC command is required to delete the MWA in this state.

Table 52. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
OLCTERMI	GBL	Online change terminate in progress. A TERMINATE OLC command is entered. Online change termination is in progress for the IMS systems in the IMSplex. Online change termination aborts the online change. If all of the IMS systems are in an online change prepare state, TERMINATE OLC aborts the online change and removes all of the IMS systems from the online change state. If an error occurs before the OLCSTAT data set is updated, then TERMINATE OLC aborts the online change. The online change abort phase is performed on the IMS systems where abort is needed. All of the IMS systems are removed from the online change state.
SECCMD	LCL	Eligible IMS commands issued from the master terminal, and their command responses, are sent to the secondary master when the command /SMC MASTER ON is issued.
SECCMDT	LCL	Eligible IMS commands that are issued from terminals other than the master terminal, and their command responses, are sent to the secondary master when the command /SMC TERMINAL ON is issued.
SECMSG	LCL	IMS system messages are sent to the secondary master when the command /SMC MESSAGE ON is issued.
XRFALT	LCL	XRF alternate system.

Output fields

The following table shows information about the QUERY MEMBER TYPE(IMS) output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 53. Output fields for QUERY MEMBER command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	N/A	Completion code for the line of output. The completion code is always returned.

Table 53. Output fields for QUERY MEMBER command (continued)

Short label	Long label	Keyword	Scope	Meaning
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. IMS identifier of IMS that built the output. The IMS identifier is always returned.
LATTR	LclAttr	ATTRIB	LCL	Local IMS attributes. See Table 51 on page 121 for more information.
MODI	ModId	MODID	LCL	Online change modify id, which is incremented by 1 for each online change.
LSTT	LclStat	STATUS	LCL	Local IMS status. See Table 52 on page 121 for more information.
STT	Status	STATUS	GBL	Global IMS status. See Table 52 on page 121 for more information.
TYP	Type	TYPE	LCL	IMSplex member type. The IMSplex member type can be IMS, for the IMS address space.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY MEMBER command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 54. Return and reason codes for QUERY MEMBER command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY MEMBER TYPE(IMS) command completed successfully.
X'00000014'	X'00005004'	The QUERY MEMBER command failed because a DFSOCMD response buffer could not be obtained.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual member.

The following table contains the completion code that can be returned on a QUERY MEMBER command.

Table 55. Completion codes for QUERY MEMBER

Completion code	Meaning
0	The QUERY MEMBER TYPE(IMS) command completed successfully for this IMS.

Examples

The following are examples of the QUERY MEMBER command:

Example 1 for QUERY MEMBER TYPE (IMS) command

TSO SPOC input:

```
QRY MEMBER TYPE(IMS) SHOW(ALL)
```

TSO SPOC output:

```
MbrName    CC Type  Status LclAttr LclStat    ModId
SYS3        0 IMS                                1
```

OM API input:

```
CMD(QRY MEMBER TYPE(IMS) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 16:32:12.998765</statime>
<stotime>2002.163 16:32:12.999775</stotime>
<staseq>B7C4B78AFD86D562</staseq>
<stoseq>B7C4B78AFDC5FA80</stoseq>
<rqsttkn1>USRT002 10093212</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>MEMBER  </kwd>
<input>QRY MEMBER TYPE(IMS) SHOW(ALL)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"   llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"    llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT" align="right" />
<hdr slbl="TYP"   llbl="Type"    scope="LCL" sort="n" key="0" scroll="yes" len="5"
dtype="CHAR" align="left" />
<hdr slbl="STT"   llbl="Status"  scope="GBL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LATTR" llbl="LclAttr" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LSTT"  llbl="LclStat" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="MODI"  llbl="ModId"   scope="LCL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" /></cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3    ) CC(   0) TYP(IMS) MODI(       1) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS member SYS3 is active in the IMSplex.

Example 2 for QUERY MEMBER TYPE(IMS) command

TSO SPOC input:

```
QRY MEMBER TYPE(IMS) SHOW(ALL)
```

TSO SPOC output:

```
MbrName    CC Type  Status LclAttr LclStat    ModId
SYS3        0 IMS                  LEOPT          1
```

OM API input:

```
CMD(QRY MEMBER TYPE(IMS) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 16:42:10.557119</statime>
<stotime>2002.163 16:42:10.557503</stotime>
<staseq>B7C4B9C4DDCBF28D</staseq>
<stoseq>B7C4B9C4DDE3F02D</stoseq>
<rqsttkn1>USRT002 10094210</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>MEMBER </kwd>
<input>QRY MEMBER TYPE(IMS) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"   llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"    llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT" align="right" />
<hdr slbl="TYP"   llbl="Type"    scope="LCL" sort="n" key="0" scroll="yes" len="5"
dtype="CHAR" align="left" />
<hdr slbl="STT"   llbl="Status"  scope="GBL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LATTR" llbl="LclAttr" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LSTT"  llbl="LclStat" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="MODI"  llbl="ModId"   scope="LCL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3    ) CC(   0) TYP(IMS) LSTT(LEOPT) MODI(       1) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS member SYS3 is active in the IMSplex. The local status shows that LE runtime option overrides are enabled.

Example 3 for QUERY MEMBER TYPE(IMS) command

TSO SPOC input:

```
QRY MEMBER TYPE(IMS) SHOW(ALL)
```

TSO SPOC output:

```
 Response for: QUERY MEMBER TYPE(IMS) SHOW(ALL)
MbrName    CC Type  Status LclAttr        LclStat    ModId
IMS2        0 IMS          SHAREDQ,GBLOLC                1
IMS3        0 IMS          SHAREDQ,GBLOLC                1
SYS3        0 IMS          SHAREDQ,GBLOLC                1
```

OM API input:

```
CMD (QRY MEMBER TYPE(IMS) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 15:13:05.255654</statime>
<stotime>2002.163 15:13:06.479196</stotime>
<staseq>B7C4A5DB308E6544</staseq>
<stoseq>B7C4A5DC5B45C385</stoseq>
<rqsttkn1>USRT011 10081304</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS3 </master>
<userid>USRT011</userid>
<verb>QRY</verb>
<kwd>MEMBER</kwd>
<input>QUERY MEMBER TYPE(IMS) SHOW(ALL)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"   llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"    llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT"  align="right" />
<hdr slbl="TYP"   llbl="Type"    scope="LCL" sort="n" key="0" scroll="yes" len="5"
dtype="CHAR" align="left" />
<hdr slbl="STT"   llbl="Status"  scope="GBL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LATTR" llbl="LclAttr" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LSTT"  llbl="LclStat" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left"/>
<hdr slbl="MODI"  llbl="ModId"   scope="LCL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right"/>
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS3    ) CC(   0) TYP(IMS) LATTR(SHAREDQ,GBLOLC) MODI(       1) </rsp>
<rsp>MBR(IMS2    ) CC(   0) TYP(IMS) LATTR(SHAREDQ,GBLOLC) MODI(       1) </rsp>
<rsp>MBR(SYS3    ) CC(   0) TYP(IMS) LATTR(SHAREDQ,GBLOLC) MODI(       1) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS members SYS3, IMS2, and IMS3 are active in the IMSplex. All three IMS systems have shared queues enabled and global online change enabled.

Example 4 for QUERY MEMBER TYPE(IMS) SHOW(STATUS) command

TSO SPOC input:

```
QUERY MEMBER TYPE(IMS) SHOW(STATUS)
```

TSO SPOC output:

```
MbrName    CC  Status                              LclStat
IMS1        0                                      OLCCMT1C,OLCCMT2I
IMS2        0                                      OLCCMT1C,OLCCMT2I
IMS2        0  OLCMSTR,OLCMACB,OLCCMT1C,OLCCMT2I
IMS3        0                                      OLCCMT1C,OLCCMT2I
```

OM API input:

```
CMD (QUERY MEMBER TYPE(IMS) SHOW(STATUS))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.275 18:28:38.671513</statime>
<stotime>2006.275 18:28:38.683275</stotime>
<staseq>BF7E87571A0999E4</staseq>
<stoseq>BF7E87571CE8B96E</stoseq>
<rqsttkn1>USRT001 10112838</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>MEMBER          </kwd>
<input>QUERY MEMBER TYPE(IMS) SHOW(STATUS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="GBL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS2    ) CC(   0) STT(OLCMACB,OLCPREPC) </rsp>
<rsp>MBR(IMS2    ) CC(   0) LSTT(OLCPREPC,SECCMD,SECMSG) </rsp>
<rsp>MBR(IMS1    ) CC(   0) LSTT(OLCPREPC,SECCMD,SECMSG) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS2 is the command master for the ACB member online change process. All IMS systems finished online change commit phase 1. The member OLC is currently in commit phase 2. The online change process cannot be terminated using a TERM OLC command. All updated members will be committed to the active ACBLIB.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY MSLINK command

Use the QUERY MSLINK command to query information about the definition and status of specified logical links.

Subsections:

- "Environment" on page 130
- "Syntax" on page 130
- "Keywords" on page 131

- "Usage notes" on page 134
- "Equivalent IMS type-1 commands" on page 134
- "Output fields" on page 134
- "Return, reason, and completion codes" on page 140
- Examples

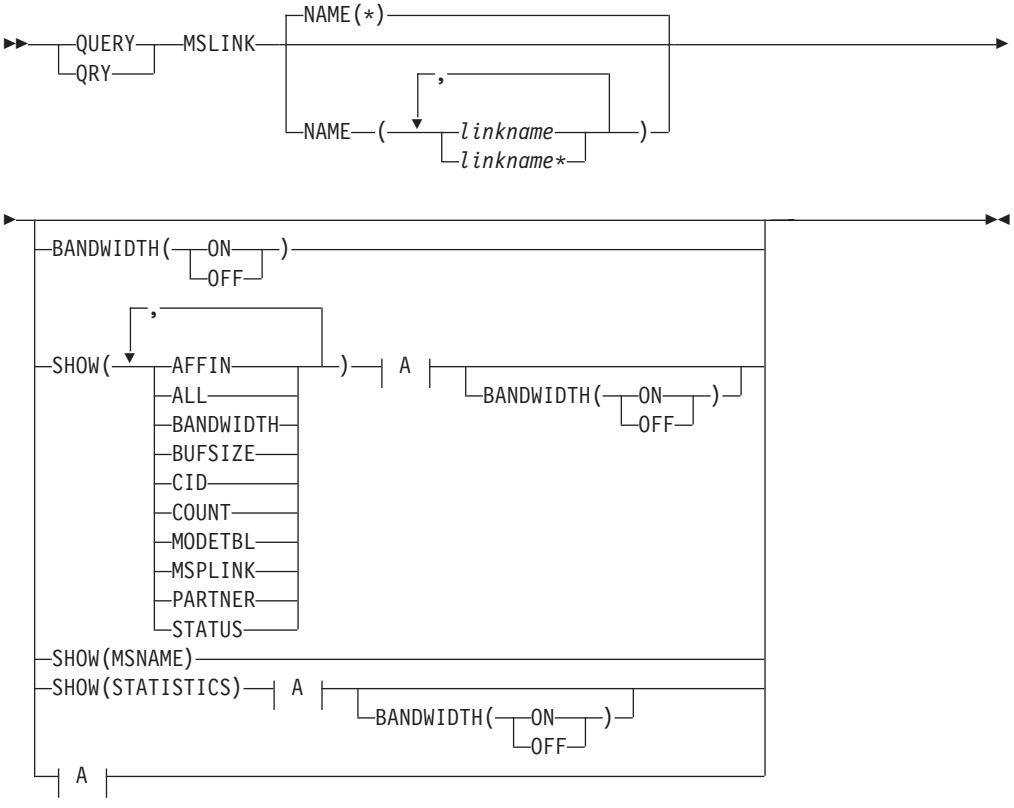
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY MSLINK command and keywords can be issued.

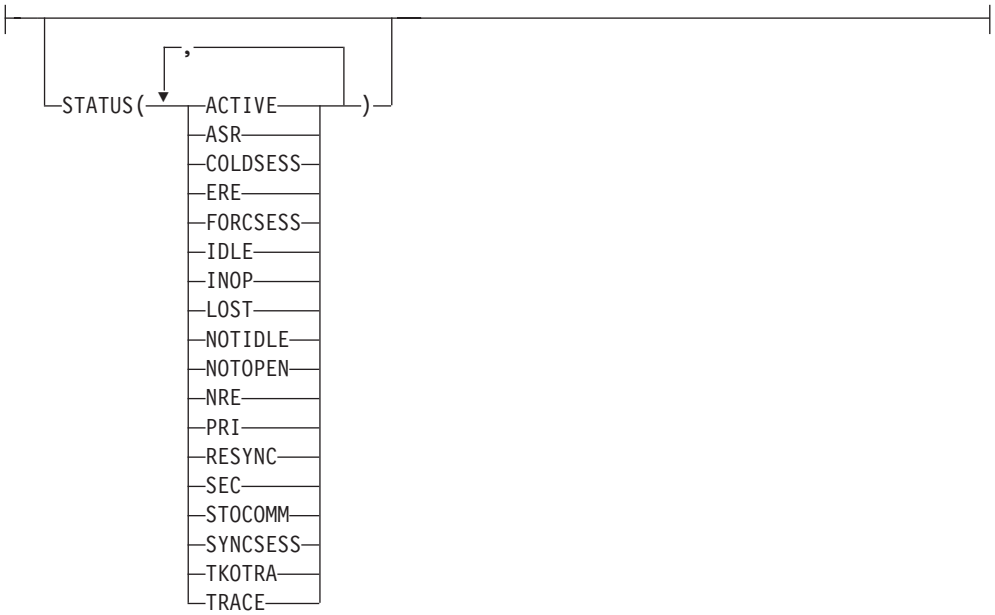
Table 56. Valid environments for the QUERY MSLINK command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
QUERY MSLINK	X		X
NAME	X		X
BANDWIDTH	X		X
SHOW	X		X
STATUS	X		X

Syntax



A:



Keywords

The following keywords are valid for the QUERY MSLINK command:

BANDWIDTH()
This filter selects logical links that are either using, or not using, bandwidth mode.

When you specify the BANDWIDTH keyword, the output returned includes the bandwidth mode, even if you do not specify SHOW(BANDWIDTH).

ON Selects only logical links that are using bandwidth mode.

OFF
Selects only logical links that are not using bandwidth mode.

NAME()
Specifies the 1- to 8-character names of the logical links displayed. NAME(*) indicates that the command is to be applied to ALL the links in the system. Wildcards (*) can be specified in the name. The default is NAME(*) which returns all MSLINK resources. The logical link number, assigned during system generation, cannot be specified. Instead, the logical link name must be specified. The logical link number is returned as command output.

SHOW()
Specifies the MSLINK output fields to be returned. If you do not specify the SHOW() parameter, only the logical link names and associated logical link numbers are returned.

The MSLINK logical link name is always returned along with the name of the IMS that created the output for the link and the completion code. The filters supported with the SHOW parameter are:

AFFIN
In an IMS system that uses VTAM Generic Resources, displays the real APPLID name in the VGR group with which the specified logical link has an affinity.

ALL
Includes all of the information in the other SHOW parameters, except for MSNAME.

BANDWIDTH
Displays whether the link is using bandwidth mode.

BUFSIZE
Displays the current input and output buffer size for the logical links.

CID
Displays the VTAM Communication Identifications (CIDs) for the logical links.

COUNT
Displays the number of messages sent to and received from the specified logical links.

MODETBL
Displays the mode table names that are associated with the specified links. The output includes DEF MODETBL and ACT MODETBL. DEF MODETBL is the default mode table name that is set by system definition or by issuing an UPDATE command. The name can be overridden with the UPDATE command or, for non-IMS session initiations, the LOGON exit. ACT MODETBL is the mode table name used to initiate the session. This name is displayed only while the session is active. The field is blank at normal session termination.

MSNAME
Displays the MSNAMEs assigned to the specified logical link, and their remote and local SYSIDs. When SHOW(MSNAME) is specified, STATUS() is invalid. MSNAME is mutually exclusive of all other SHOW filters.

MSPLINK
Displays the physical link to which this logical link is assigned.

PARTNER
Displays the current partner ID for this logical link.

STATISTICS
Displays the statistics for the logical link.

MSC link statistics are displayed by type-2 command QUERY MSLINK NAME(linkname) SHOW(STATISTICS). The SHOW(STATISTICS) keyword also displays the statistics reset mode, RESET,CHKPT or NORESET,CHKPT. Link statistics are not displayed with the SHOW(ALL) keyword.

I/O statistics are recorded only in bandwidth mode.

STATUS
Displays the logical link status.

STATUS()
Displays logical links that possess at least one of the specified status. When you specify the STATUS keyword, the output returned includes the logical link status even if you do not specify SHOW(STATUS). The STATUS keyword cannot be specified with SHOW(MSNAME).

ACTIVE
Indicates, for the logical link, that link startup processing is complete and the line is available for message transfer.

ASR
Indicates the current automatic session restart designation as part of the Session Outage Notification.

COLDSESS
Indicates that for the logical link, startup processing is not complete.

ERE
Indicates that for the logical link, startup processing is not complete. However, when started, emergency restart synchronization is performed, because the previous link shutdown was either not normal or an IMS emergency restart was performed.

FORCSESS
Indicates that message resynchronization is attempted when the logical link is started. Even if resynchronization fails, the link is still started.

IDLE
Indicates that no activity is in progress for the logical link.

INOP
Indicates an inoperable link.

LOST
Indicates that the VTAM LOSTERM EXIT has been scheduled for this link.

NOTIDLE
Indicates that the link is waiting for the completion of a synchronous event. The status can be NOTIDLE-A, NOTIDLE-B, or NOTIDLE-C*xx*, where *xx* represents the value of the access method operation code.

NOTIDLE-A
Indicates a status of NOTIDLE-POST, which means that an event has completed but the link has not been dispatched to process it.

NOTIDLE-B
Indicates a status of NOTIDLE-IWAIT, which means the link is waiting for completion of internal I/O.

NOTIDLE-C*xx*
Indicates a status of NOTIDLE-TP WAIT, which means the link is waiting for completion of a TP access method request. The two characters *xx* indicate the value of the access method operation code. Blanks appear if the link is VTAM.

NOTOPEN
Indicates that the link is not in open status.

NRE
Indicates that startup processing is not complete for the logical link. Normal restart synchronization is performed when the logical link starts, because the previous link shutdown or IMS restart was normal.

PRI
Indicates that the link is the primary partner of the MSC VTAM session.

RESYNC
Indicates that the positive acknowledgment for an IMS recoverable output message was not received when the MSC session was terminated.

SEC
Indicates that the link is the primary partner of the MSC VTAM session.

STOCOMM
Indicates that the link was stopped for communications.

SYNCSESS
Indicates that message resynchronization is attempted when the logical link is started, and that the link is started only if resynchronization is successful.

TKOTRA
Indicates that the logical link in an XRF session is traced only during takeover, to help diagnose XRF link switch problems.

TRACE
Indicates that the logical link is traced.

Usage notes

You can issue this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY MSLINK command and the type-1 IMS commands that perform similar functions.

Table 57. Type-1 equivalents for the QUERY MSLINK command

QUERY MSLINK command	Similar IMS type-1 command
QUERY MSLINK NAME(linkname | linkname*) SHOW(ALL)	/DIS LINK link# | ALL
QUERY MSLINK NAME(linkname | linkname*) SHOW(MODETBL)	/DIS LINK link# | ALL MODE
QUERY MSLINK NAME(linkname | linkname*) SHOW(MSPLINK|MSNAME)	/DIS ASSIGNMENT LINK link# | ALL
QUERY MSLINK NAME(linkname | linkname*) SHOW(AFFIN)	/DIS AFFIN LINK *link*# | ALL /DIS AFFIN NODE *nodename*

Output fields

The query output will display a series of headers and values, including the MSLINK name and number, IMSID that processed the command, and the command condition code (CC) of the command.

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Following are the statistics headers on the QUERY MSLINK command, and their meanings. All count fields with times and rates are in seconds. For example, 1.3 is equal to 1.3 seconds. .000003 is equal to 3 microseconds. The other fields such as byte or message counts are in decimal numbers. Rates are in units per second (for example, CHKW_RATE means check write calls per second).

Table 58. Output fields for the QUERY MSLINK command

Short label	Long label	Keyword	Meaning
AFFIN	Affin	AFFIN	In an IMS system using VTAM Generic Resources, this field displays the real APPLID name in the VGR group with which the specified logical link has an affinity.
AMRS	Avg_Msg_Rec_SZ	STATISTICS	Average message size received (type 01/03 message record).
AMSS	Avg_Msg_Send_SZ	STATISTICS	Average message size sent (type 01/03 message record).
AMTB	ActMdtbl	MODETBL	Active VTAM logon mode table entry for the logical link.
APT	Avg_Proc_Time	STATISTICS	Average link processing time per dispatch calculated by dividing TPT by TDN.
AQGT	Avg_Qget_Time	STATISTICS	The average time for QMGR calls (GU or DEQ) to process a send message.
AQPT	Avg_Qput_Time	STATISTICS	The average QMGR call (ISRT or ENQ) to process a received message.
BANDW	Bandwidth	BANDW	The current usage of bandwidth mode.
BSR	Send_MsgByte_Rate	STATISTICS	Message bytes sent per second from MSC link. The send time per each message sent is the time from Get Unique (GU) to get the message to send, to dequeue the message when the response is received. The link idle time in between sending messages is ignored.
BUFSZ	BufSize	BUFSIZE	Input and output buffer size for the logical link.
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
CID	CID	CID	VTAM Communication Identification.

Table 58. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Meaning
CNTR	RecdCnt	COUNT	Number of messages received from the specified link.
CNTS	SentCnt	COUNT	Number of messages sent on the specified link.
CWION	ChkwIO_CT	STATISTICS	Number of the logger check writes that resulted in a write (I/O) of the get unique (GU) or insert (ISRT) call to the logger write ahead data set (WADS).
CWN	Chkw_CT	STATISTICS	Number of logger check writes. Check writes are requests to the logger to insure the message is logged (for example, Message get uniques are logged on the send side and inserts are logged on the receive side).
CWR	Chkw_Rate	STATISTICS	Logger check write rate calculated by dividing the CWR by the statistics recording time.
DMTB	DefMdtbl	MODETBL	Default VTAM logon mode table entry (set by system definition or modified by the UPDATE command).
HMRS	Hi_Msg_Rec_SZ	STATISTICS	Largest message size received (type 01/03 message record).
HMSS	Hi_Msg_Send_SZ	STATISTICS	Largest message size sent (type 01/03 message record).
HPT	Hi_Proc_Time	STATISTICS	Highest (longest) time the link was dispatched to process.
HQGT	Hi_Qget_Time	STATISTICS	The highest (longest) QMGR call (GU or DEQ) to process a send message.
HQPT	Hi_Qput_Time	STATISTICS	The highest (longest) QMGR call (ISRT or ENQ) to process a received message.
HRIOT	Hi_RecIO_Time	STATISTICS	The highest (longest) I/O time to receive a message.
HSIOT	Hi_SentIO_Time	STATISTICS	The highest (longest) I/O time to send a message.
LINKN	MSLink#	N/A	Logical link number.
LMRS	Low_Msg_Rec_SZ	STATISTICS	Smallest message size received (type 01/03 message record).
LMSS	Low_Msg_Send_SZ	STATISTICS	Smallest message size sent (type 01/03 message record).
LPT	Low_Proc_Time	STATISTICS	Lowest (shortest) time the link was dispatched to process.
LQGT	Low_Qget_Time	STATISTICS	The lowest (shortest) QMGR call (GU or DEQ) to process a send message.

Table 58. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Meaning
LQPT	Low_Qput_Time	STATISTICS	The lowest (shortest) QMGR call (ISRT or ENQ) to process a received message.
LRIOT	Low_RecIO_Time	STATISTICS	The lowest (shortest) I/O time to receive a message.
LSIOT	Low_SendIO_Time	STATISTICS	The lowest (shortest) I/O time to send a message.
LSTT	LclStat	STATUS	The current status of the logical link. For the possible status values returned, see the description of the STATUS keyword.
MBR	MbrName	N/A	IMSplex member that built the output line.
MSL	MSLink	N/A	Logical link name.
MSN	MSName	MSNAME	MSNAMEs associated with the logical link.
MSP	MSPLink	MSPLINK	Physical link to which the logical link is assigned.
MSR	Send_MsgCT_Rate	STATISTICS	Messages sent per second by MSC link. The send time per each message sent is the time from Get Unique (GU) to get the message to send, to dequeue the message when the response is received. The link idle time in between sending messages is ignored.
PID	PID	PARTNER	Partner ID for the link.
RBR	RecIO_Byte_Rate	STATISTICS	Rate in bytes per second being received. This is calculated by dividing the total bytes received (TBRC) by the total send time (TRIOT).
RIOR	RecIO_Req_Rate	STATISTICS	The receive I/O requests per second. This is calculated by dividing the total receives (TRN) by the total send I/O time (TRIOT).
RMT	Rec_Msg_Time	STATISTICS	Interval of time between the first and last message received. Used for benchmark testing of a block of messages.
SBR	SendIO_Byte_Rate	STATISTICS	Rate in bytes per second being sent. This is calculated by dividing the total bytes sent (TBSC) by the total send time (TSIOT).
SIDL	SIDL	MSNAME	Local system identification of the associated MSNAME.
SIDR	SIDR	MSNAME	Remote system identification of the associated MSNAME.

Table 58. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Meaning
SIOR	SendIO_Req_Rate	STATISTICS	The send I/O requests per second. This is calculated by dividing the total sends (TSN) by the total send I/O time (TSIOT).
SMT	Send_Msg_Time	STATISTICS	Interval of time between the first and last message sent. Used for benchmark testing of a block of messages.
SOPT	Option	STATISTICS	Statistics reset option = RESET,CHKPT = statistics are reset at each IMS checkpoint or NORESET,CHKPT = statistics are not reset at IMS checkpoints.
STIM	Start_Time	STATISTICS	The start date and local time for the statistics. For option RESET,CHKPT, this will be the last IMS checkpoint time. For option NORESET,CHKPT, this will be when the last UPDATE MSLINK START STATISTICS OPTION (RESET) command was issued. The format is yyyy.ddd hh.mm.ss.tt, where: **yyyy** Four digit year **ddd** Julian day of the year **hh** hour of the day, 00 - 23 **mm** minute of the hour, 00 - 59 **ss** second of the minute, 00 - 59 **tt** tenths of the second, 0 - 9 **hh** hundredths of the second, 0 - 9
TBRC	Tot_Byte_Rec_CT	STATISTICS	Total bytes of data received including message data, response data, and internal message control blocks used for bandwidth mode.
TBSC	Tot_Byte_Send_CT	STATISTICS	Total bytes of data sent including message data, response data, and internal message control blocks used for bandwidth mode.
TDN	Tot_Disp_CT	STATISTICS	Number of times the logical link ITASK was dispatched. A typical send or receive message operation will require two dispatches, one to send and one to process the acknowledgment, or one to receive and one to send the acknowledgment.

Table 58. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Meaning
TMBRC	Tot_MsgByte_Rec_CT	STATISTICS	Total count of message bytes of data received. This includes the message prefixes and user data segments (for example, all the data in the type 01/03 log record).
TMBSC	Tot_MsgByte_Send_CT	STATISTICS	Total count of message bytes of data sent. This includes the message prefixes and user data segments (for example, all the data in the type 01/03 log record).
TMRN	Tot_Msg_Rec_CT	STATISTICS	Total number of messages received.
TMSN	Tot_Msg_Send_CT	STATISTICS	Total count of messages sent.
TPT	Tot_Proc_Time	STATISTICS	Total amount of time the logical link ITASK was dispatched to be processed. This includes wait time while the ITASK was dispatched. This time is in seconds with a resolution to micro seconds.
TQGN	Tot_Qget_CT	STATISTICS	Total count of QMGR calls issued to get messages off the queue (local or shared queues) for processing. This includes get unique (GU) and dequeue (DEQ) calls.
TQGT	Tot_Qget_Time	STATISTICS	Total processing time for QMGR calls issued to get messages off the queue (local or shared queues) for processing. This includes get unique (GU) and dequeue (DEQ) calls. Times are calculated from the time the call was issued by the MSC ITASK, to the time the call returned from QMGR.
TQPN	Tot_Qput_CT	STATISTICS	Total count of QMGR calls issued to put messages to the queue (local or shared queues) for processing. This includes insert (ISRT) and enqueue (ENQ) calls.
TQPT	Tot_Qput_Time	STATISTICS	Total processing time for QMGR calls issued to put messages to the queue (local or shared queues) for processing. This includes insert (ISRT) and enqueue (ENQ) calls. Times are calculated from the time the call was issued by the MSC ITASK, to the time the call returned from QMGR.

Table 58. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Meaning
TRIOT	Tot_RecIO_Time	STATISTICS	Total amount of I/O time to receive messages. This is time from which the access method was called to process the receive, to the time the partner IMS access method was called to send the message. This is the time between the access method calls on each side, for the data to cross the media (for example, CTC link, memory, or network).
TRN	Tot_Req_CT	STATISTICS	Total count of receive I/O requests received from the access method.
TSIOT	Tot_SendIO_Time	STATISTICS	Total amount of I/O time to send messages. This is the time from which the access method was called to process the send, to the time the partner IMS ITASK was dispatched upon receipt of the message. This is the time between the access method calls on each side, for the data to cross the media (for example, CTC link, memory, or network).
TSN	Tot_Send_CT	STATISTICS	Total count of send I/O requests issued to the access method.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY MSLINK command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 59. Return and reason codes for the QUERY MSLINK command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY MSLINK command completed successfully.
X'00000008'	X'00002040'	More than one filter value is specified on the QUERY MSLINK command.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. For details on completion codes, see Table 60 on page 141.

Table 59. Return and reason codes for the QUERY MSLINK command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. For details on completion codes, see Table 60.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000014'	X'00005004'	The QUERY MSLINK command processing terminated as a DFSOCMD response buffer could not be obtained.

Errors that are unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 60. Completion codes for the QUERY MSLINK command

Completion code	Completion code text	Meaning
0		The QUERY MSLINK command completed successfully for the resource.
10	NO RESOURCES FOUND	MSLINK name is invalid, or the specified wildcard parameter does not match any resource names.

Example 1 for QUERY MSLINK command

TSO SPOC input:

```
QRY MSLINK NAME(STAR*) SHOW(ALL)
```

TSO SPOC output:

(screen 1)

```
MSLink      MSLink# MbrName CC MSPLink CID        PID   RecdCnt SentCnt
STAR1L            1  IMSA    0  STAR1   20003BAA1L            0       6
STAR3L1           4  IMSA    0  STAR3B1 20003BAB3L           15      33
STAR3L2           5  IMSA    0  STAR3B1 20003BAC3L            0       0
STAR3L3           6  IMSA    0  STAR3B1 20003BAD3L            5       5
STARZZZ          11  IMSA    0                                0       0
```

(scrolled to the right, screen 2)

```
MSLink  MSLink# MbrName DefMdtbl ActMdtbl BufSize Bandwidth Affin LclStat
STAR1L       1  IMSA                       65536   Y       IMSA  ACTIVE,ASR
STAR3L1      4  IMSA    LOGON13B            4096   Y       IMSA  ACTIVE
STAR3L2      5  IMSA    LOGON13B            4096   Y             IDLE
STAR3L3      6  IMSA    LOGON13B            4096   Y             LOST
STARZZZ     11  IMSA                        1024   N             IDLE
```

Explanation: All logical links matching STAR* are returned with all output fields. The MSNAME information is not returned with SHOW(ALL).

Example 2: SHOW(MSNAME)

TSO SPOC input:

```
QUERY MSLINK NAME(STAR*) SHOW(MSNAME)
```

TSO SPOC output:

```
MSLink   MSLink# MbrName CC MSName  SIDR SIDL
STAR1L         1  IMSA    0  STARN1   15   35
STAR1L         1  IMSA    0  STARN11  14   34
STAR3L1        4  IMSA    0  STARN31  41   51
STAR3L2        5  IMSA    0  STARN32 318  438
STAR3L3        6  IMSA    0  STARN33   3   13
STARZZZ       11  IMSA    0
```

Explanation: The QUERY MSLINK SHOW(MSNAME) command returns the MSNAMEs and the SYSID information for all the MSNAMEs that are associated with the logical link.

Example 3: Querying statistics for VTAM links

Explanation: In this example, the QUERY MSLINK command is issued twice: once in the IMS systems at each end of the VTAM link. The commands were issued after each side sent 10 remote transactions that processed and sent a response message back (10 transactions + 10 responses = 20 messages sent each direction).

TSO SPOC input:

```
QUERY MSLINK NAME(LNK12V02) SHOW(STATISTICS)
```

TSO SPOC output:

(screen 1)

```
MSLink    MSLink#  MbrName   CC    Option           Start_Time
LNK12V02       10     IMS1    0    NORESET,CHKPT    2006.261 19:03:58.77
```

(scrolled to the right, screen 2)

```
MSLink    MSLink#  MbrName    Tot_Disp_CT  Tot_Proc_Time  Hi_Proc_Time  Low_Proc_Time  Avg_Proc_Time
LNK12V02       10     IMS1             98       0.176661      0.059530       0.000003       0.000853
```

(scrolled to the right, screen 3)

```
MSLink    MSLink#  MbrName    Chkw_CT   ChkwIO_CT  Chkw_Rate  Tot_Send_CT  Tot_Msg_Send_CT
LNK12V02       10     IMS1         33          33   0.165016           26               20
```

(scrolled to the right, screen 4)

```
MSLink    MSLink#  MbrName    Tot_MsgByte_Send_CT  Tot_Byte_Send_CT  Hi_Msg_Send_SZ  Low_Msg_Send_SZ
LNK12V02       10     IMS1                10,461            22,403             578              502
```

(scrolled to the right, screen 5)

```
MSLink    MSLink#  MbrName    Avg_Msg_Send_SZ  Send_Msg_Time  Tot_Qget_CT  Tot_Qget_Time  Hi_Qget_Time
LNK12V02       10     IMS1               523       2.408548           51       0.002054      0.001070
```

(scrolled to the right, screen 6)

```
MSLink    MSLink#  MbrName    Low_Qget_Time  Avg_Qget_Time  Tot_SendIO_Time  Hi_SendIO_Time
LNK12V02       10     IMS1         0.000004       0.000040         0.505401        0.193435
```

(scrolled to the right, screen 7)

```
MSLink    MSLink#  MbrName    Low_SendIO_Time  SendIO_Req_Rate  SendIO_Byte_Rate  Tot_Rec_CT
LNK12V02       10     IMS1           0.000708        51.444298            44,327          26
```

(scrolled to the right, screen 8)

```
MSLink    MSLink#  MbrName    Tot_Msg_Rec_CT  Tot_MsgByte_Rec_CT  Tot_Byte_Rec_CT  Hi_Msg_Rec_SZ
LNK12V02       10     IMS1               20              10,503           22,445            584
```

(scrolled to the right, screen 9)

```
MSLink    MSLink#  MbrName    Low_Msg_Rec_SZ  Avg_Msg_Rec_SZ  Rec_Msg_Time  Tot_Qput_CT  Tot_Qput_Time
LNK12V02       10     IMS1              502             525      2.320062           40       0.037326
```

(scrolled to the right, screen 10)

```
MSLink    MSLink#  MbrName    Hi_Qput_Time  Low_Qput_Time  Avg_Qput_Time  Tot_RecIO_Time  Hi_RecIO_Time
LNK12V02       10     IMS1         0.015957       0.000008       0.000933        0.070947       0.025376
```

(scrolled to the right, screen 11)

```
MSLink    MSLink#  MbrName    Low_RecIO_Time  RecIO_Req_Rate  RecIO_Byte_Rate
LNK12V02       10     IMS1          0.000155      366.470750       148,040.09
```

TSO SPOC input:

```
QUERY MSLINK NAME(LNK21V02) SHOW(STATISTICS)
```

TSO SPOC output:

(scrolled to the right, screen 1)

```
MSLink    MSLink#  MbrName   CC    Option            Start_Time
LNK21V02       13  IMS2       0    NORESET,CHKPT     2006.261 19:03:08.42
```

(scrolled to the right, screen 2)

```
MSLink    MSLink#  MbrName    Tot_Disp_CT  Tot_Proc_Time  Hi_Proc_Time  Low_Proc_Time  Avg_Proc_Time
LNK21V02       13  IMS2               97       0.150468      0.060122       0.000003       0.000696
```

(scrolled to the right, screen 3)

```
MSLink    MSLink#  MbrName    Chkw_CT   ChkwIO_CT  Chkw_Rate  Tot_Send_CT  Tot_Msg_Send_CT
LNK21V02       13  IMS2            33          33   0.544554           26               20
```

(scrolled to the right, screen 4)

```
MSLink    MSLink#  MbrName    Tot_MsgByte_Send_CT  Tot_Byte_Send_CT  Hi_Msg_Send_SZ  Low_Msg_Send_SZ
LNK21V02       13  IMS2                   10,503            22,445             584              502
```

(scrolled to the right, screen 5)

```
MSLink    MSLink#  MbrName    Avg_Msg_Send_SZ  Send_Msg_Time  Tot_Qget_CT  Tot_Qget_Time  Hi_Qget_Time
LNK21V02       13  IMS2                  525       2.395508           51       0.003884      0.001169
```

(scrolled to the right, screen 6)

```
MSLink    MSLink#  MbrName    Low_Qget_Time  Avg_Qget_Time  Tot_SendIO_Time  Hi_SendIO_Time
LNK21V02       13  IMS2            0.000004       0.000076         1.304330        0.217838
```

(scrolled to the right, screen 7)

```
MSLink    MSLink#  MbrName    Low_SendIO_Time  SendIO_Req_Rate  SendIO_Byte_Rate  Tot_Rec_CT
LNK21V02       13  IMS2             0.000821        19.933605            17,208          26
```

(scrolled to the right, screen 8)

```
MSLink    MSLink#  MbrName    Tot_Msg_Rec_CT  Tot_MsgByte_Rec_CT  Tot_Byte_Rec_CT  Hi_Msg_Rec_SZ
LNK21V02       13  IMS2                20              10,461           22,403            578
```

(scrolled to the right, screen 9)

```
MSLink    MSLink#  MbrName    Low_Msg_Rec_SZ  Avg_Msg_Rec_SZ  Rec_Msg_Time  Tot_Qput_CT  Tot_Qput_Time
LNK21V02       13  IMS2                502             523      2.344878           40       0.054777
```

(scrolled to the right, screen 10)

```
MSLink    MSLink#  MbrName    Hi_Qput_Time  Low_Qput_Time  Avg_Qput_Time  Tot_RecIO_Time  Hi_RecIO_Time
LNK21V02       13  IMS2          0.023559       0.000007       0.001369        0.210252       0.036846
```

(scrolled to the right, screen 11)

```
MSLink    MSLink#  MbrName    Low_RecIO_Time  RecIO_Req_Rate  RecIO_Byte_Rate
LNK21V02       13  IMS2             0.000155      123.661130          106,553
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

List of commands with similar functions for multiple resources (Operations and Automation)

QUERY MSNAME command

Use the QUERY MSNAME command to query definition and status information about the specified logical link path.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 145
- "Usage notes" on page 147
- "Equivalent IMS type-1 commands" on page 147
- "Output fields" on page 147
- "Return, reason, and completion codes" on page 148
- "Example" on page 150

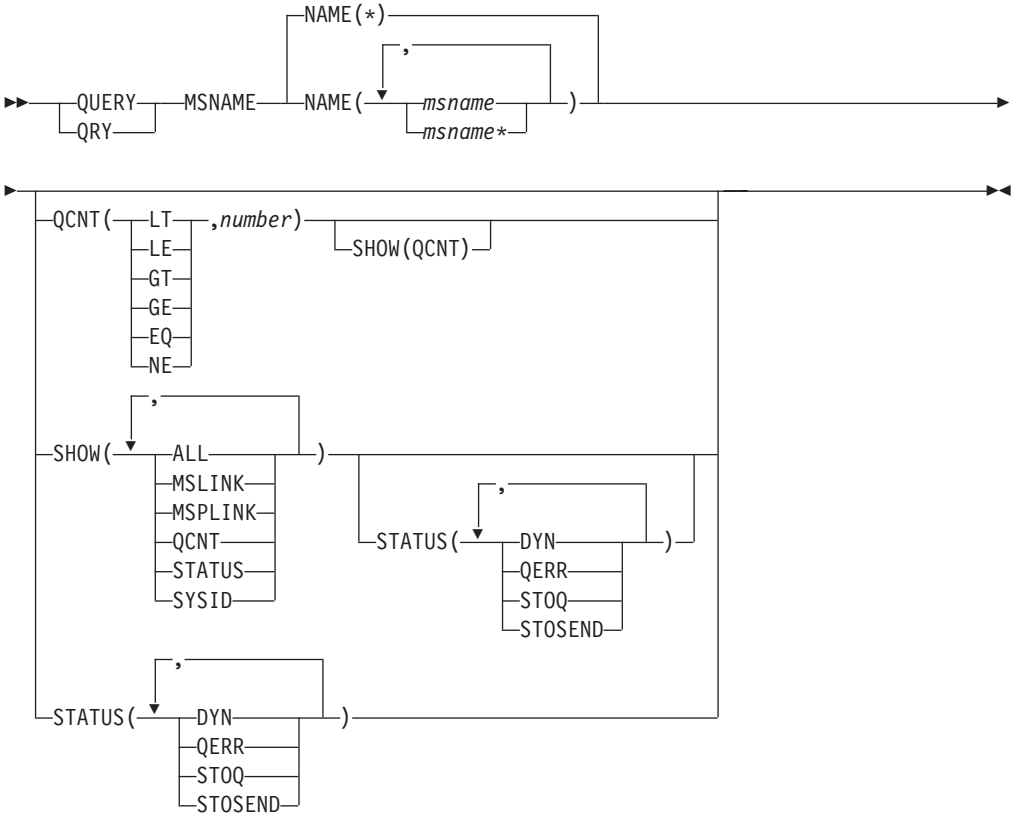
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY MSNAME command and keywords can be issued.

Table 61. Valid environments for the QUERY MSNAME command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
QUERY MSNAME	X		X
NAME	X		X
QCNT	X		X
STATUS	X		X
SHOW	X		X

Syntax



Keywords

NAME()

Specifies the 1- to 8-character names of the MSC logical link paths that are to be displayed or indicates that the command is to be applied to all the links in the system. Wildcards (*) can be specified in the name. The default is NAME(*), which returns all MSNAME resources.

QCNT()

Selects logical link paths that have a queue count less than (LT), less than or equal to (LE), greater than (GT), greater than or equal to (GE), equal to (EQ), or not equal to (NE) the specified numbers. The specified number cannot be a 1 when LT is specified. This allows additional filtering by QCNT value.

Logical link paths with a queue count of 0 are not returned when you specify the QCNT parameter. When you specify the QCNT(LT,n) parameter, *msnames* with a queue count greater than 0 or less than 'n' are returned. If you do not specify the STATUS or QCNT parameters, all of the logical link paths that match the MSNAME names are returned.

The QCNT filter is valid in both a shared-queues environment and a non-shared-queues environment.

In a shared-queues environment, if QCNT is specified, the shared queues are slower than local queues. In this environment, the QRY MSNAME command is processed only by the master IMS because the queues are global. The command master returns all the logical link paths on the shared queues that match the specified queue count parameter. If QCNT is specified with a

wildcard logical link path name, all of the shared queues logical link path messages on the coupling facility must be read.

In a non-shared-queues environment, the local queue count values are used to determine the *msnames* to be displayed. In this environment, the QRY MSNAME QCNT command is processed by each IMS that the command is routed to because the queues are local. Each IMS returns all the *msnames* found locally that match the specified queue count parameter.

When you specify the QCNT parameter, the output returned includes the queue count, even if the SHOW(QCNT) option is not specified.

The QCNT filter is mutually exclusive with the STATUS filter. The QCNT filter is mutually exclusive with SHOW(), except for SHOW(QCNT).

SHOW()
Specifies the MSNAME output fields to be returned. If SHOW() is not specified, only the logical link path names and associated remote system identification and local system identification are returned. The MSNAME name is always returned, along with the name of the IMS that created the output for the link path and the completion code. Only SHOW(QCNT) is allowed with the QCNT() parameter. All other SHOW filters are mutually exclusive with the QCNT() parameter. The parameters supported with the SHOW keyword are:

ALL
Includes all of the information in the other SHOW parameters.

MSLINK
Displays the logical link assigned to the specified logical link path.

MSPLINK
Displays the physical link that is assigned to the specified logical link path.

QCNT
Specifies that queue count information is returned. Both local and global queue counts are returned.

The local queue counts value returned on this command represents the messages being processed by the IMS system where this command is issued. In a shared-queues environment, do not use the local queue counts for queue evaluation purposes. The global queue counts, which are returned on this command, represent the current message queue counts on the shared queues structure.

In a shared-queues environment, the command master returns both the global and its local queue count information for all the MSNAMEs specified. The MSNAME does not have to be defined at the command master IMS. All other IMS systems return their local queue count information.

In a non-shared-queues environment, all IMS systems return their local queue count information.

STATUS
Displays the logical link path status.

SYSID
Displays the remote system identification and the local system identification for this logical link path.

STATUS()
Displays logical link paths that display at least one of the specified status. The STATUS filter is mutually exclusive with the QCNT filter. When you specify

the STATUS keyword, the output returned includes the logical link path status, even if you do not specify SHOW(STATUS).

DYN
Displays MSNAMEs that have been dynamically created in a shared queue environment.

QERR
Displays MSNAMEs with the queue error status.

STOQ
Displays MSNAMEs with the stopped queuing status.

STOSEND
Displays MSNAMEs with the stopped sending status.

Usage notes

You can issue this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY MSNAME and the type-1 IMS commands that perform similar functions.

Table 62. Type-1 equivalents for the QUERY MSNAME command

QUERY MSNAME command	Similar IMS type-1 command
QUERY MSNAME NAME(msname | msname*) SHOW(ALL)	/DIS MSNAME msname | msname*
QUERY MSNAME NAME(msname | msname*) SHOW(QCNT)	/DIS MSNAME msname QCNT
QUERY MSNAME NAME(*) SHOW(ALL)	/DIS MSNAME ALL
QUERY MSNAME NAME(msname | *) SHOW(MSPLINK | MSLINK)	/DIS ASSIGNMENT MSNAME msname | msname* | ALL

Output fields

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 63. Output field descriptions for the QUERY MSNAME command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
LINKN	MSLink#	MSLINK	Link number of logical link associated with the link path.
LQ	LQCnt	QCNT	Local queue count.
LSTT	LclStat	STATUS	The current status of the logical link path.
MBR	MbrName	N/A	IMSplex member that built the output line.
MSL	MSLink	MSLINK	Logical link associated with the logical link path.
MSN	MSName	N/A	Logical link path name.
MSP	MSPLink	MSPLINK	Physical link associated with the logical link path.
QCNT	QCnt	QCNT	Global queue count.
SIDL	SIDL	SYSID	Local system identification for the logical link path.
SIDR	SIDR	SYSID	Remote system identification for the logical link path.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY MSNAME command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 64. Return and reason codes for the QUERY MSNAME command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY MSNAME command completed successfully.
X'00000008'	X'00002040'	More than one filter value is specified on the QUERY MSNAME command.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.

Table 64. Return and reason codes for the QUERY MSNAME command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'00004004'	IMS attempted to obtain the message queue count from CQS, but CQS was not active.
X'00000010'	X'00004005'	IMS attempted to obtain the message queue count from CQS, but CQS was not connected to the queue structure.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000014'	X'00005004'	The QUERY MSNAME command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005104'	IMS attempted to obtain the message queue count from CQS, but CQS returned with an unexpected return code.
X'00000014'	X'00005FFF'	IMS attempted to obtain the message queue count from CQS, but failed because of an internal IMS error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 65. Completion codes for the QUERY MSNAME command

Completion code	Completion code text	Meaning
0		The QUERY MSNAME command completed successfully for the resource.
10	NO RESOURCES FOUND	MSNAME name is invalid, or the specified wildcard parameter does not match any resource names.

Table 65. Completion codes for the QUERY MSNAME command (continued)

Completion code	Completion code text	Meaning
98	CQS REQUEST ERROR	IMS was unable to obtain the message queue count from CQS.

Example

In this example, the specified logical link path is returned with all output fields.

TSO SPOC input:

```
QRY MSNAME NAME(STAR1N1,STAR1N2) SHOW(ALL)
```

TSO SPOC output:

(screen 1)

```
MSName  MBRname CC Qcnt MSPLink MSLink  MSLink# SIDR SIDL
STAR1N1 IMSA    0  765
STAR1N1 IMSA    0       STAR1   STAR1L        1   10   20
STAR1N2 IMSA    0    0
STAR1N2 IMSA    0       STAR1   STAR1L1       2   11   21
```

(scrolled to the right, screen 2)

```
MSName  MBRname LQCnt LclStat
STAR1N1 IMSA
STAR1N1 IMSA    0
STAR1N2 IMSA
STAR1N2 IMSA    0
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

List of commands with similar functions for multiple resources (Operations and Automation)

QUERY MSPLINK command

Use the QUERY MSPLINK command to query definition and status information about the specified physical links.

Subsections:

- "Environment" on page 151
- "Syntax" on page 151
- "Keywords" on page 151
- "Usage notes" on page 153
- "Equivalent IMS type-1 commands" on page 153
- "Output fields" on page 153
- "Return, reason, and completion codes" on page 154
- Examples

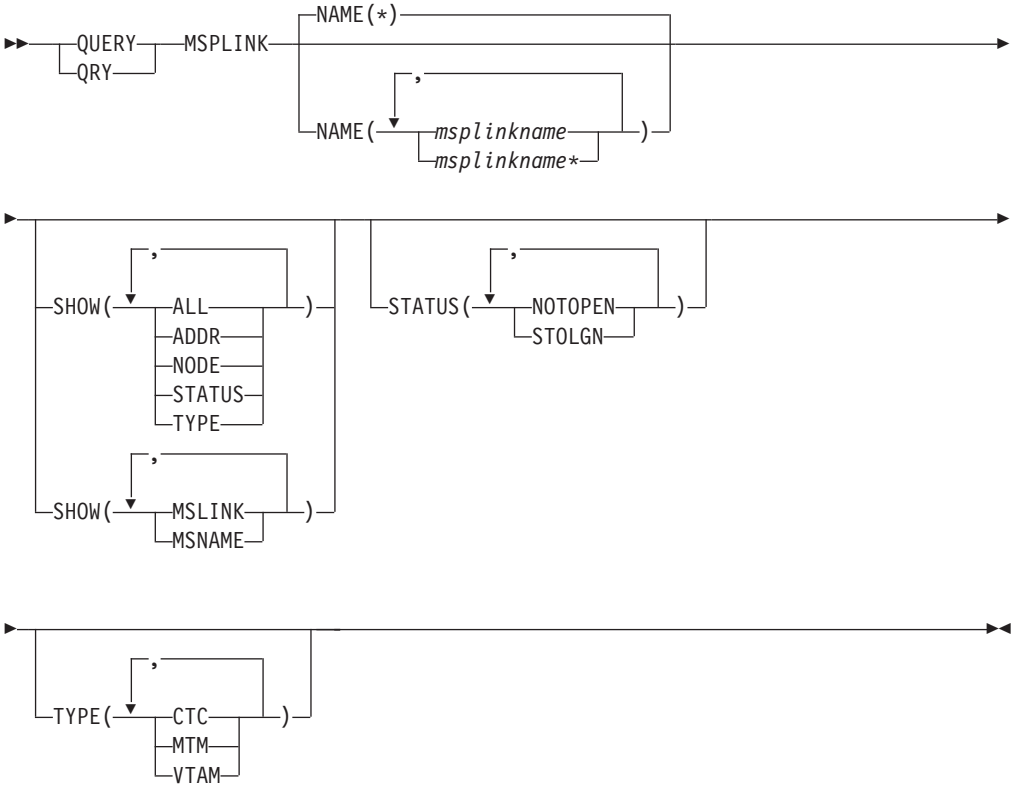
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY MSPLINK command and keywords can be issued.

Table 66. Valid environments for the QUERY MSPLINK command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
QUERY MSPLINK	X		X
NAME	X		X
SHOW	X		X
STATUS	X		X
TYPE	X		X

Syntax



Keywords

The following keywords are valid for the QUERY MSPLINK command:

NAME()
Specifies the 1- to 8-character names of the physical links for which physical link information is to be returned. Wildcards (*) can be specified in the name. The default is NAME(*), which returns all MSPLINK resources.

SHOW()
Specifies the MSPLINK output fields to be returned. If SHOW() is not specified, only the physical names and associated link types are returned.

The MSPLINK name is always returned along with the name of the IMS that created the output for the link and the completion code. The parameters supported with the SHOW keyword are:

ALL
Includes all of the information in the other SHOW parameters (except MSLINK and MSNAME).

ADDR
Displays the address of the channel-to-channel adapter for CTC physical links.

MSLINK
Displays the logical links that are associated with the specified physical link. When the physical link is VTAM, the maximum number of allowed logical sessions for the physical link is also displayed. When SHOW(MSLINK) is specified, STATUS() is invalid. SHOW(MSNAME) and SHOW(MSLINK) are mutually exclusive with all other SHOW parameters.

MSNAME
Displays the logical link paths that are associated with the specified physical link. The MSNAME names, and the remote and local system identifications are displayed. When SHOW(MSNAME) is specified, STATUS() is invalid. SHOW(MSNAME) and SHOW(MSLINK) are mutually exclusive with all other SHOW parameters.

NODE
Displays the VTAM node name (APPLID) of the remote systems at the other end of the physical links for VTAM physical links.

STATUS
Displays the status of the physical link.

TYPE
Displays the type of physical link.

STATUS()
Displays physical links that display at least one of the specified statuses. You cannot specify the STATUS() parameter with the SHOW(MSLINK) parameter or the SHOW(MSNAME) parameter. When you specify the STATUS keyword, the output returned includes the physical link status, even if you do not specify SHOW(STATUS).

NOTOPEN
VTAM ACB has not been opened.

STOLGN
Displays physical link that specify the IMS ID of this IMS system have been stopped.

TYPE()
Displays physical links that possess at least one of the specified types of physical links. When you specify the TYPE keyword, the output returned includes the type of physical link, even if you do not specify SHOW(TYPE).

CTC
Channel-to-Channel adapter.

MTM
Memory-to-Memory.

VTAM
Virtual Telecommunication Access Method.

Usage notes

You can specify this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY MSPLINK command and the type-1 IMS commands that perform similar functions.

Table 67. Type-1 equivalents for the QUERY MSPLINK command

QUERY MSPLINK command	Similar IMS type-1 command
QUERY MSPLINK NAME(msplinkname | *) SHOW(ALL)	No similar type-1 IMS command exists.
QUERY MSPLINK NAME(msplinkname | *) SHOW(MSLINK | MSNAME)	/DIS ASSIGNMENT MSPLINK msplinkname | ALL

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 68. Output field descriptions for the QUERY MSPLINK command

Short label	Keyword	Meaning
ADDR	ADDR	Address of the communication line or CTC adapter.
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
LINKN	MSLINK	Link number of logical link associated with the physical link.

Table 68. Output field descriptions for the QUERY MSPLINK command (continued)

Short label	Keyword	Meaning
LSTT	STATUS	Status of the physical link, including automatic session restart option.
MAXS	MSLINK	Maximum number of sessions allowed.
MBR	N/A	IMSplex member that built the output line.
MSL	MSLINK	Logical links associated with the physical link.
MSN	MSNAME	MSNAMEs associated with the physical link.
MSP	N/A	Physical link name.
NODE	NODE	VTAM node name of the remote system at the other end of the physical link.
SIDL	MSNAME	Local system identification of the associated MSNAME.
SIDR	MSNAME	Remote system identification of the associated MSNAME.
TYPE	TYPE	Type of the physical link: VATM, CTC, or MTM.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY MSPLINK command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 69. Return and reason codes for the QUERY MSPLINK command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY MSPLINK command completed successfully.
X'00000008'	X'00002040'	More than 1 filter value is specified on the QUERY MSPLINK command.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.

Table 69. Return and reason codes for the QUERY MSPLINK command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000014'	X'00005004'	The QUERY MSPLINK command processing terminated because a DFSOCMD response buffer could not be obtained.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 70. Completion codes for the QUERY MSPLINK command

Completion code	Completion code text	Meaning
0		The QUERY MSPLINK command completed successfully for the resource.
10	NO RESOURCES FOUND	MSPLINK name is invalid, or the specified wildcard parameter does not match any resource names.

Example 1 for QUERY MSPLINK command

In this example, the specified physical links with names that begin with STAR1 are returned with all output fields.

TSO SPOC input:

```
QRY MSPLINK NAME(STAR1*) SHOW(ALL)
```

TSO SPOC output:

```
MSPLink  MbrName CC Type NodeName CTCaddr LclStat
STAR1    IMSA     0 VTAM APPL12B
STAR1B11 IMSA     0 VTAM APPL11B
STAR1B12 IMSA     0 VTAM APPL14B
STAR1B13 IMSA     0 VTAM APPL13B
```

Example 2 for QUERY MSPLINK command

In this example, the specified physical links with names that begin with STAR1 are returned with `MSLink` and `MSName` output fields. Each physical link is VTAM, so each one displays an output line containing the maximum allowable sessions.

TSO SPOC input:

```
QRY MSPLINK NAME(STAR1*) SHOW(MSLINK,MSNAME)
```

TSO SPOC output:

```
MSPLink MbrName CC MSLink MSLink# MaxSess MSName SIDR SIDL
STAR1    IMSA   0                      2
STAR1    IMSA   0  STAR1L       1         STARN1   15   35
STAR1    IMSA   0  STAR1L       1         STARN11  14   34
STAR1B11 IMSA   0                      2
STAR1B11 IMSA   0
STAR1B12 IMSA   0                      3
STAR1B12 IMSA   0
STAR1B13 IMSA   0                      3
STAR1B13 IMSA   0  STAR3L1      4         STARN31  41   51
STAR1B13 IMSA   0  STAR3L2      5         STARN32 318  438
STAR1B13 IMSA   0  STAR3L3      6         STARN33   3   13
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

List of commands with similar functions for multiple resources (Operations and Automation)

QUERY NODE command

Use the QUERY NODE command to display information about VTAM nodes (terminals) and non-VTAM devices (SPOOL and SYSOUT devices) across the IMSplex.

This command can be specified only through the OM API and is valid on an XRF alternate.

Subsections:

- "Environment" on page 157
- "Syntax" on page 157
- "Keywords" on page 159
- "Usage notes" on page 164
- "Equivalent IMS type-1 commands" on page 165
- "Output fields" on page 166
- "QUERY NODE status" on page 171
- "Return, reason, and completion codes" on page 173
- "Examples" on page 175

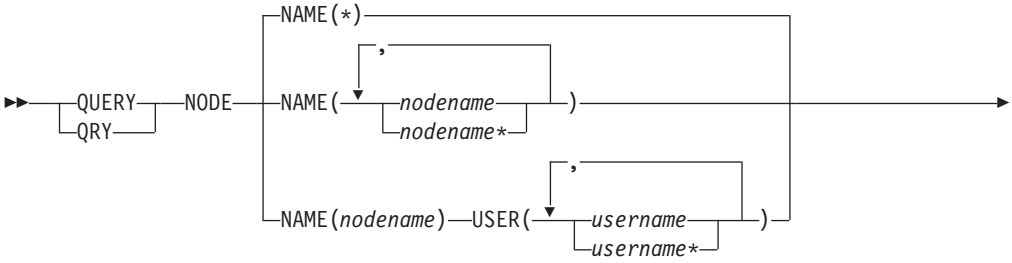
Environment

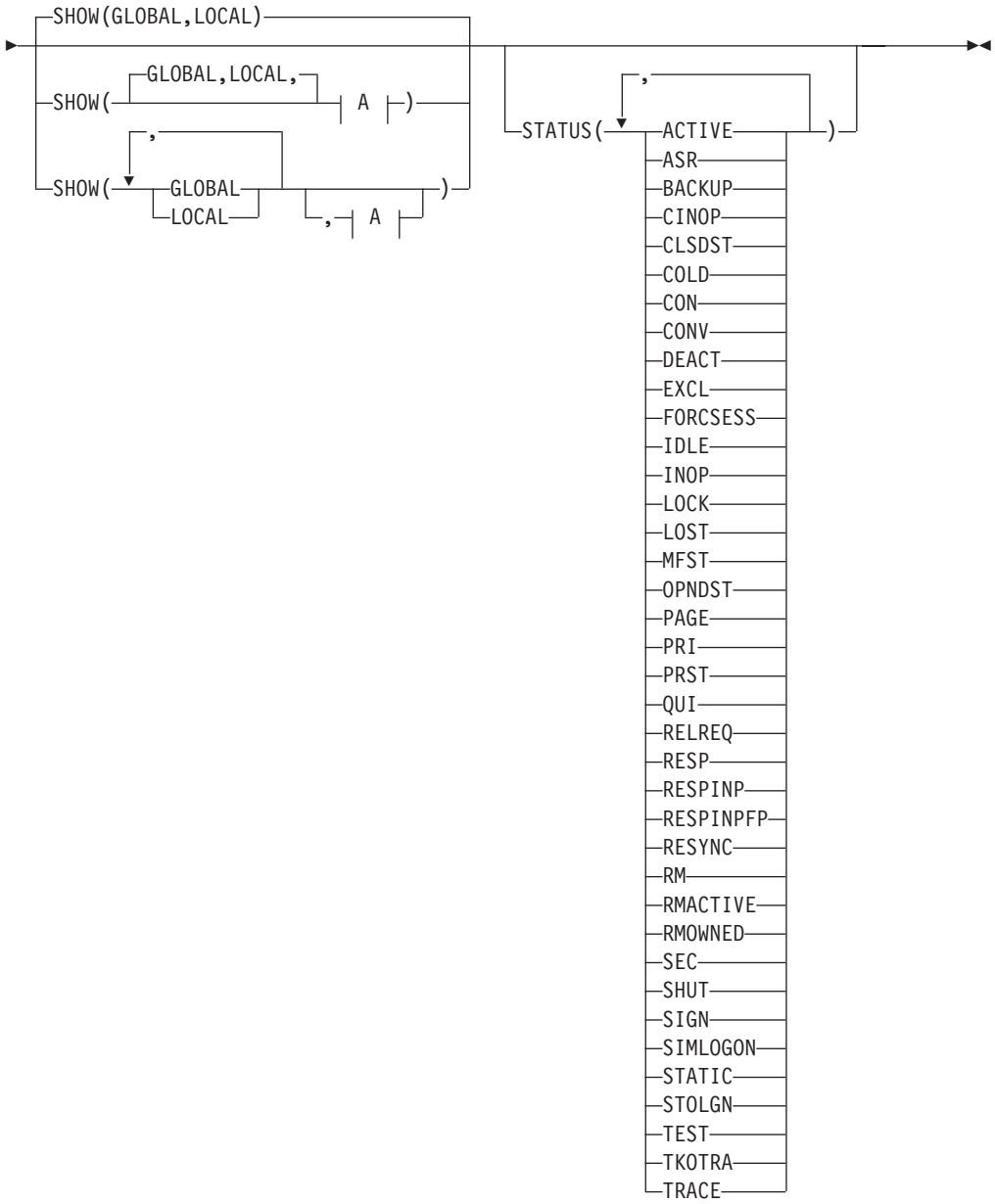
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 71. Valid environments for the QUERY NODE command and keywords

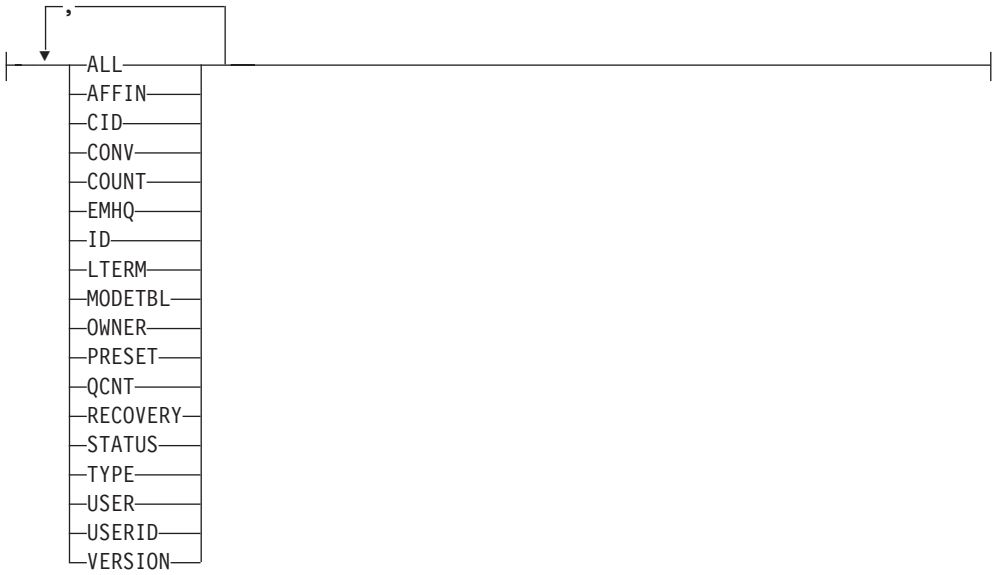
Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY NODE	X		X
NAME	X		X
USER	X		X
SHOW	X		X
STATUS	X		X

Syntax




SHOW(GLOBAL,LOCAL)
SHOW(
GLOBAL,LOCAL,
A
)
SHOW(
,
GLOBAL
LOCAL
,
A
)
STATUS(
,
ACTIVE
ASR
BACKUP
CINOP
CLSDST
COLD
CON
CONV
DEACT
EXCL
FORCSESS
IDLE
INOP
LOCK
LOST
MFST
OPNDST
PAGE
PRI
PRST
QUI
RELREQ
RESP
RESPINP
RESPINPFP
RESYNC
RM
RMACTIVE
RMOWNED
SEC
SHUT
SIGN
SIMLOGON
STATIC
STOLGN
TEST
TKOTRA
TRACE
)

A:



Keywords

The following keywords are valid for the QUERY NODE command:

NAME()
Specifies the names of one or more VTAM nodes (terminals) or non-VTAM devices that are to be displayed. Valid names are 1-8 characters, and wildcards can be specified. To display all nodes and devices, specify NAME(*). NAME(*) is the default.

Any static or dynamic VTAM terminal name can be specified. Supported terminal types are LU0, LU1, LU2, and LU6.1 (ISC only).

Any non-VTAM device can be specified. Non-VTAM devices are assigned a LINE and PTERM number during the system definition process. Supported device types are CONSOLE, SYSOUT (DISK, PUNCH, PRINTER, READER, TAPE), SPOOL, and TCO:

CONSOLE
This is the system console. IMS automatically assigns LINE 1 PTERM 1 to this device.

DISK Defined by the LINEGRP macro, UNITYPE=DISK.

PUNCH
Defined by the LINEGRP macro, UNITYPE=PUNCH.

PRINTER
Defined by the LINEGRP macro, UNITYPE=PRINTER.

READER
Defined by the LINEGRP macro, UNITYPE=READER.

SPOOL
Defined by the LINEGRP macro, UNITYPE=SPOOL.

TAPE Defined by the LINEGRP macro, UNITYPE=TAPE.

TCO This is the device assigned as the Time Control Option. The system definition process automatically assigns the last LINE number to this device.

To specify a non-VTAM device, specify a name of DFSLN*xxx*, where *xxx* is the LINE number of the device. For example, to display the system console (LINE 1), specify NAME(DFSLN001).

SHOW()
Specifies the node output fields to be returned.

The node name is always returned along with the name of the IMS that created the output and the completion code. If SHOW is not specified, only the node names are returned, provided that the STATUS filter is not specified. This provides a method for a system management application to obtain a list of all node names that are currently known in the IMSplex.

Two parameters, GLOBAL and LOCAL, are used to specify the location (global resources or local resources) where IMS should obtain the information that is to be displayed. The default is both GLOBAL and LOCAL.

The rest of the parameters are used to specify what information is displayed.

The parameters supported with the SHOW keyword, which can be specified in any order, are:

ALL
Returns all of the output fields, except for those fields displayed when the LTERM and CONV parameters are specified. To display LTERM and conversation information with all other output, specify SHOW(ALL,LTERM,CONV).

AFFIN
IMS APPLID to which the node has VTAM generic resource affinity, if applicable. VTAM generic resource affinity is valid only when the GLOBAL parameter is specified on the SHOW keyword. If GLOBAL is not specified, then the AFFIN parameter is ignored.

CID
VTAM connection identifier.

CONV
The conversation ID, transaction, and conversation status associated with the node. Each conversation is returned on a separate command response display line. Conversation status is not included when SHOW(ALL) is specified. To display conversation information with all other output, specify SHOW(ALL,CONV).

COUNT
Displays the number of messages sent to and received from the specified node.

EMHQ
Displays the node message queue count in the Expedited Message Handler (EMH) queues. The queue count is the sum of the queue counts for each logical terminal associated with the specified node.

EMHQ is valid only when the GLOBAL parameter is specified on the SHOW keyword. If GLOBAL is not specified, then the EMHQ parameter is ignored.

EMHQ is processed by the command master only. It is ignored by all other IMS systems.

EMHQ is valid only if shared EMH is used in a shared queues environment. Otherwise, this parameter is ignored.

If the node resource exists in the resource structure, then the logical terminals associated with the global node resource are used to obtain the queue counts from the EMH queues. Otherwise, the logical terminals associated with the local node resource are used, but if the node does not exist locally on the command master, then the queue count is 0.

GLOBAL
If GLOBAL is specified, the command master displays global information, depending on the other SHOW parameters specified. This includes information from the resource structure.

The GLOBAL parameter is processed by the command master only. All other IMS systems ignore this parameter. If LOCAL is not also specified, then all IMS systems other than the command master ignore the command.

GLOBAL is applicable only when the command master is using shared queues, VTAM generic resources, sysplex terminal management, or any combination of these. GLOBAL is not applicable when the command master is not using shared queues, VTAM generic resources, or sysplex terminal management. In this environment, if LOCAL is also specified, then GLOBAL is ignored. Otherwise, the command master rejects the command.

If shared queues are enabled, and global queue counts are requested, then the command master will make requests to CQS to determine the appropriate queue counts. This includes both MSGQ and EMHQ.

If VTAM generic resources (VGR) is enabled, and VGR affinity information is requested, then the command master will make requests to VTAM to determine any VGR affinity.

If sysplex terminal management is enabled, then the command master will make requests to RM to determine the appropriate global status.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

ID Displays the other half-session qualifier name of the ISC node.

LOCAL
If LOCAL is specified, then all IMS systems including the command master display local information, depending on the other SHOW parameters specified. This includes information local to the IMS processing the command.

The LOCAL parameter is processed by all IMS systems, including the command master. LOCAL is applicable in any environment, regardless of whether shared queues or sysplex terminal management are used.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

LTERM
Displays the logical terminal (LTERM) names, if any, associated with the node. A node may have zero or more logical terminals associated with it.

Each logical terminal associated with the node is returned on a separate command response line.

The LTERM status is not included when SHOW(ALL) is specified. To display logical terminal information with all other output, specify SHOW(ALL,LTERM).

MODETBL
Displays the mode table names that are associated with the specified nodes. The output includes both the default mode table name and the actual mode table name. The default mode table name is the default name that is set by system definition. This name can be modified by the UPDATE NODE command, or by the LOGON exit. The actual mode table name is the actual name used to initiate the session. This name is displayed only while the session is active, and is blank at all other times.

OWNER
Displays the owner of the node resource in the resource structure. This applies only when sysplex terminal management is enabled, and is only processed by the command master. All other IMS systems ignore this parameter.

The owner is the IMSID (or RSENAME for XRF systems) of the IMS system that owns the node. An IMS system owns a node resource if the resource is active (the node is logged on), or an IMS system is maintaining significant status for that resource.

PRESET
Displays the preset destination name for a node. A node is in preset destination mode set by the /SET command. The preset destination name is either a transaction name, or a logical terminal (LTERM) name. All messages entered from this node are sent to the preset destination transaction or LTERM.

QCNT
Displays the node message queue count. The queue count is the sum of the queue counts for each logical terminal associated with the specified node.

The local queue counts value returned on this command represents the messages being processed by the IMS system where this command is issued. In a shared-queues environment, do not use the local queue counts for queue evaluation purposes. The global queue counts, which are returned on this command, represent the current message queue counts on the shared queues structure.

If the LOCAL parameter is also specified on the SHOW keyword, then all IMS systems that process the command, including the command master, display the local queue count. This is valid whether or not shared message queues are enabled.

If the GLOBAL parameter is also specified on the SHOW keyword, and shared message queues are enabled, then the command master displays the global queue count on the shared message queues (MSGQ). If the node resource exists in the resource structure, then the logical terminals associated with the global node resource are used to obtain the queue counts from shared queues. Otherwise, the logical terminals associated with the local node resource are used, but if the node does not exist locally on the command master, then the queue count is 0.

The local and global queue counts are displayed as separate output fields.

RECOVERY
Displays the status recovery mode (SRM) and level of recovery for the node. End-user significant status can be conversation, Fast Path, full-function response mode, or STSN (set-and-test-sequence-number) status.

SRM determines where end-user significant status, if any exists, is recovered following a session or IMS termination. The output displays the SRM for the node as either GLOBAL (sysplex terminal management recovers it in the resource structure), LOCAL (IMS recovers it locally, which indicates an affinity to a particular IMS), or NONE (status is discarded).

Level of recovery determines what end-user significant status, if any exists, is recovered (if SRM is GLOBAL or LOCAL). The output displays whether conversation status is recovered (RCVYCONV), Fast Path status is recovered (RCVYFP), full-function response mode is recovered (RCVYRESP), and STSN status is recovered (RCVYSTSN).

STATUS
Returns local or global status of the node. See "QUERY NODE status" on page 171 for a list and meaning of possible status that might be returned.

TYPE
VTAM node type. The possible node types are AVM, FIN, LUT6, NTO, SLUP, SLU1, SLU2, 3277, 3286, 3790, CONSOLE, RDR/PTR (for SYSOUT and SPOOL), and TCO.

USER
Displays the dynamic or ISC user associated with the node. For ISC parallel-session nodes, this is the user structure representing the half-session qualifier. For static ISC, this is defined with the SUBPOOL macro. For non-ISC dynamic nodes, this is the user structure allocated to the node during signon of a user ID to the node. The user name might or might not be the same as the user ID, depending on installation-specific user exits or descriptors used. Static non-ISC nodes (and single-session ISC nodes) do not have users (even if a user ID is signed on).

USERID
Displays the user ID (for security, usually RACF), signed on to the node. This is applicable to any device type, static or dynamic, and is distinct from the user name.

VERSION
Displays the version number assigned to RM resources, which is assigned by MVS, and maintained by RM, when the resources are created or updated in the resource structure.

Static single-session nodes (not parallel-session ISC) are represented in RM by two resources: the node resource, and the static-node-user resource. The VERSION parameter displays the version number for each of these resources. Static parallel-session ISC nodes and all dynamic nodes are represented in RM by one resource: the node resource. The VERSION parameter displays the version number of the node resource.

VERSION applies only when sysplex terminal management is enabled. VERSION is ignored when sysplex terminal management is not enabled.

STATUS()
Selects nodes for display that possess at least one of the specified node statuses. The status might exist locally or globally if sysplex terminal management (STM) is enabled.

The STATUS filter is valid in both a sysplex terminal management environment and in a non-sysplex terminal management environment. In a sysplex terminal management environment, the status selected might exist locally, globally, or both. If sysplex terminal management is not enabled, then the status only exists locally.

If SHOW(LOCAL) is specified, then IMS will select only those nodes with the appropriate status in the local system. The command is processed by all IMS systems, including the command master.

If SHOW(GLOBAL) is specified, and sysplex terminal management is enabled, then IMS will select only those nodes with the appropriate status in the resource structure. The command is processed only by the command master.

If SHOW(GLOBAL) is specified, but sysplex terminal management is not enabled, then the command is rejected.

If SHOW(GLOBAL,LOCAL) is specified (the default), then IMS will select those nodes with the appropriate status either locally or in the resource structure (if sysplex terminal management is enabled). The command is processed by all IMS systems. The command master processes both global and local information.

The output returned when the status filter is specified includes the status of the node, even if SHOW(STATUS) is not specified.

See "QUERY NODE status" on page 171 to determine which filters can be used to select nodes with corresponding status.

USER()
For a specific ISC node, specifies the names of one or more ISC user names allocated to the node. Valid names are 1-8 characters, and wildcards can be specified. To display all users allocated to the node, specify USER(*), or omit the USER keyword.

The output returned includes the user name even if SHOW(USER) is not specified.

Usage notes

QUERY NODE can be specified only through the OM API.

QUERY NODE can be issued on an XRF alternate system, but SHOW(GLOBAL) is not supported. Only local information can be displayed.

The processing of the QUERY NODE command is different depending on whether IMS sysplex terminal management is enabled.

- If IMS sysplex terminal management is not enabled, processing is local for each system. The results of type-1 and type-2 commands are similar.
- If IMS sysplex terminal management is enabled, type-1 and type-2 command processing is similar when displaying local information. However, they differ in how global information is displayed.
- For type-1 /DISPLAY commands with IMS sysplex terminal management enabled, the command master displays information from either the resource structure or the local system, but not both. If the resource being displayed is not

owned by any system or is owned by the command master, the command master displays the global resource. However, if the resource is owned by a system other than the command master, the command master displays only the local resource, and the owning system is responsible for displaying the global resource.

- For type-2 QUERY commands with IMS sysplex terminal management enabled, the command master is the only system that displays global resource information, regardless of whether the resource is owned. In addition, the command master displays local resource information. All other IMS systems that process the command display local resource information only. This approach allows more flexibility in displaying all information in an IMSplex.

The SHOW keyword determines which IMS systems process the command and what information is displayed.

- If SHOW(GLOBAL) is specified, then the command master displays global information, which includes the global queue count if shared queues are enabled, generic resource affinity information if VTAM generic resource (VGR) is enabled, and status from the resource structure if sysplex terminal management is enabled (STM=YES defined in DFSDCxxx PROCLIB member). This is true whether or not the node is active on any particular IMS system. All other IMS systems to which OM routes the command ignore the GLOBAL parameter with return code X'00000004' and reason code X'00001000'.
- If SHOW(LOCAL) is specified, then each IMS system to which OM routes the command (including the command master) processes the command, and displays information that is local to each system.
- If both GLOBAL and LOCAL are specified (which is the default), then the command master displays both global and local information, and all other IMS systems to which OM routes the command displays local information.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY NODE command and the type-1 IMS commands that perform similar functions.

Table 72. Type-1 equivalents for the QUERY NODE command

QUERY NODE command	Similar IMS type-1 command
QUERY NODE SHOW(AFFIN)	/DISPLAY AFFIN NODE node
QUERY NODE SHOW(CID)	/DISPLAY NODE node
QUERY NODE SHOW(CONV)	/DISPLAY CONV NODE node
QUERY NODE SHOW(COUNT)	/DISPLAY NODE node
QUERY NODE SHOW(EMHQ)	/DISPLAY NODE node QCNT EMHQ
QUERY NODE SHOW(LTERM)	/DISPLAY ASMT NODE node
QUERY NODE SHOW(MODETBL)	/DISPLAY NODE node MODE
QUERY NODE SHOW(OWNER)	/DISPLAY NODE node RECOVERY
QUERY NODE SHOW(PRESET)	/DISPLAY NODE node
QUERY NODE SHOW(QCNT)	/DISPLAY NODE node /DISPLAY NODE node QCNT
QUERY NODE SHOW(RECOVERY)	/DISPLAY NODE node RECOVERY
QUERY NODE SHOW(STATUS)	/DISPLAY NODE node
QUERY NODE SHOW(TYPE)	/DISPLAY NODE node

Table 72. Type-1 equivalents for the QUERY NODE command (continued)

QUERY NODE command	Similar IMS type-1 command
QUERY NODE SHOW(USERID)	/DISPLAY NODE node
QUERY NODE SHOW(USER)	/DISPLAY NODE node /DISPLAY ASMT NODE node
QUERY NODE STATUS(CONV)	/DISPLAY CONV
QUERY NODE STATUS(TRACE)	/DISPLAY TRACE NODE
QUERY NODE STATUS(status)	/DISPLAY STATUS NODE

Output fields

The following table shows the QUERY NODE output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

SHOW parameter
Identifies the parameter on the SHOW keyword that caused the field to be generated. Error appears for output fields that are returned for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field. GBL indicates that the field can be generated only by the command master when displaying global information for SHOW(GLOBAL). LCL indicates that the field can be generated by any IMS displaying local information for SHOW(LOCAL). N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 73. Output fields for the QUERY NODE command

Short label	Long label	SHOW parameter	Scope	Meaning
AFFIN	Affin	AFFIN	GBL	IMS APPLID to which the node has VTAM generic resource affinity, if applicable.
AMTB	ActMdtbl	MODETBL	LCL	Active mode table. This name is only displayed when the session is active.
CC	CC	N/A	N/A	Completion code. The completion code indicates whether or not IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 173 for more information. The completion code is always returned.
CCTXT	CCText	*Error*	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
CID	CID	CID	LCL	VTAM connection identifier.

Table 73. Output fields for the QUERY NODE command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
CNTR	RecdCnt	COUNT	LCL	Number of messages received from the specified node.
CNTS	SentCnt	COUNT	LCL	Number of messages sent to the specified node.
CONVID	ConvID	CONV	GBL	The conversation ID for a conversation associated with the node, as it exists in the resource structure. A node might have zero, one, or more conversations. Each conversation will have its own line of output.
CONVSTT	ConvStat	CONV	GBL	The status of a conversation associated with the node, as it exists in the resource structure. A node might have zero, one, or more conversations. The status can be: • CONVHELD - conversation is held • CONVACTV - conversation is active • CONVSCHD - conversation is scheduled
CONVTRN	ConvTran	CONV	GBL	The transaction for a conversation associated with the node, as it exists in the resource structure. A node might have zero, one, or more conversations.
DMTB	DefMdtbl	MODETBL	LCL	Default mode table.
EMHQ	EMHQCnt	EMHQ	GBL	Global logical terminal queue count in the EMH (Expedited Message Handler) queues. EMHQ is displayed only if shared EMH is used.
GBL	Gbl	GLOBAL	GBL	If 'Y', then the output reflects the status found globally in RM. If blank, then the output reflects the status found locally.
ID	ID	ID	GBL	For ISC parallel-session terminals, displays the global half-session qualifier of the other system.
ISCUSER	ISCUser	N/A	N/A	When the node is parallel-session ISC, the user subpool name is returned to differentiate parallel sessions for the same node. This field will appear when an ISC node has been selected for display. When the user name is also displayed, both the ISCUSER and USER fields will contain the same user name. For LOCAL scope, N/A is displayed for a parallel session that is not yet allocated. For GLOBAL scope, N/A is displayed once for each node, which represents global status for the node.
LCONVID	LConvID	CONV	LCL	The conversation ID for a conversation associated with the node, as it exists in the local system. A node might have zero, one, or more conversations. Each conversation will have its own line of output.

Table 73. Output fields for the QUERY NODE command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LCONVSTT	LConvStat	CONV	LCL	The status of a conversation associated with the node, as it exists in the local system. A node might have zero, one, or more conversations. The status can be: • CONVHELD: Conversation is held • CONVACTV: Conversation is active • CONVSCHD: Conversation is scheduled
LCONVTRN	LConvTran	CONV	LCL	The transaction for a conversation associated with the node, as it exists in the local system. A node might have zero, one, or more conversations.
LID	LID	ID	LCL	For ISC VTAM parallel-session terminals, displays the local half-session qualifier of the other system.
LINE	Line	N/A	LCL	Identifies the line number for system console, SPOOL, SYSOUT, or TCO device. This is only displayed if the node is non-VTAM.
LLTERM	LLterm	LTERM	LCL	Local logical terminal names. The logical terminal names associated with the node.
LPRST	LPreset	PRESET	LCL	Identifies the preset destination transaction or LTERM name when the node is in preset destination mode, which is established by the /SET command. All messages entered at this terminal are sent to the preset destination trancode or LTERM.
LQ	LQCnt	QCNT	LCL	Local queue count.
LRCVY	LRcvy	RECOVERY	LCL	The level of recovery for end-user significant status in the local system, which indicates what type of status is recoverable. Any value presented here implies that the corresponding status is recoverable. If SRM is LOCAL, the status will be recovered locally. If SRM is GLOBAL, the status will be recovered globally. These values are not applicable if SRM is NONE or there is no SRM. The status values that can be returned (more than one are possible) are: • CONV: IMS conversations are recoverable (RCVYCONV=YES). • FP: Fast Path status is recoverable (RCVYFP=YES). • RESP: Full-function response mode status is recoverable (RCVYRESP=YES). • STSN: STSN status is recoverable (RCVYSTSN=YES).

Table 73. Output fields for the QUERY NODE command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LSRM	LSRM	RECOVERY	LCL	The status recovery mode in the local system, which determines where the end-user significant status is maintained and recovered from. The output will be one of the following: • GBL: Status is saved globally in the IMS resource structure. • LCL: Status is saved in local control blocks and log records. • NONE: Status is not saved in the IMS resource structure or log records. • Blank: SRM is not yet established, the node is not logged on and there is no end-user significant status, or there is no user signed on.
LSTT	LclStat	STATUS	LCL	Local node status. See "QUERY NODE status" on page 171 for a list and explanation of the possible node status.
LTERM	Lterm	LTERM	GBL	Global logical terminal names. The logical terminal names associated with the node.
LTYPE	LType	TYPE	LCL	VTAM node type. The possible node types are: AVM, FIN, LUT6, NTO, SLUP, SLU1, SLU2, 3277, 3286, 3790, CONSOLE, RDR/PTR (for SYSOUT and SPOOL), and TCO.
LUID	LUserid	USERID	LCL	Identifies the local user ID signed on to the node.
LUSER	LUser	USER	LCL	Identifies the local dynamic or ISC user allocated to the node.
LVER	LVersion#	VERSION	LCL	Version number for the node resource being maintained in the local system. This field applies only when STM is enabled.
LVERSNU	LVersion#SNU	VERSION	LCL	Version number for the static-node-user resource being maintained in the local system. This field applies only to static single-session nodes when STM is enabled.
MBR	MbrName	N/A	N/A	IMSplex member (modular unit) that built the output line. IMS identifier of the IMS that built the output. The IMS identifier is always returned.
NODE	Node	N/A	N/A	The node name. The node name is always returned.
OWNER	Owner	OWNER	GBL	Resource owner. IMS identifier or RSENAME of IMS where the node is active. If no owning IMS system exists and RM contains an entry for the resource, the owner field will be blank.
QCNT	QCnt	QCNT	GBL	Global queue count on the shared queues. Global queue count can be displayed only if shared queues are used.
PTERM	PTerm	N/A	LCL	Identifies the PTERM number for system console, SPOOL, SYSOUT, or TCO device. This is only displayed if the node is non-VTAM.

Table 73. Output fields for the QUERY NODE command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
RCVY	Rcvy	RECOVERY	GBL	The level of recovery for end-user significant status in the resource structure, which indicates what type of status is recoverable. Any value presented here implies that the corresponding status is recoverable. If SRM is LOCAL, the status will be recovered locally. If SRM is GLOBAL, the status will be recovered globally. These values are not applicable if SRM is NONE or there is no SRM. The status values that can be returned (more than one are possible) are: • CONV: IMS conversations are recoverable (RCVYCONV=YES). • STSN: STSN status is recoverable (RCVYSTSN=YES). • FP: Fast Path status is recoverable (RCVYFP=YES).
SRM	SRM	RECOVERY	GBL	The status recovery mode in the resource structure, which determines where the end-user significant status is maintained and recovered from. The output will be one of the following: • GBL: Status is saved globally in the IMS resource structure. • LCL: Status is saved in local control blocks and log records. • NONE: Status is not saved in the IMS resource structure or log records. • Blank: SRM is not yet established, the node is not logged on and there is no end-user significant status, or there is no user signed on.
STT	Status	STATUS	GBL	Global node status. See “QUERY NODE status” on page 171 for a list and explanation of the possible node status.
TYPE	Type	TYPE	GBL	VTAM node type. The possible node types are AVM, FIN, LUT6, NTO, SLUP, SLU1, SLU2, 3277, 3286, 3790.
UID	Userid	USERID	GBL	Identifies the RACF user ID signed on to the node.
USER	User	USER	GBL	Identifies the dynamic or ISC user signed on to or associated with the node.
VER	Version#	VERSION	GBL	Version number for the node resource being maintained in the resource structure. This field applies only when STM is enabled.
VERSNU	Version#SNU	VERSION	GBL	Version number for the static-node-user resource being maintained in the resource structure. This field applies only to static single-session nodes when STM is enabled.

QUERY NODE status

The following table shows the possible node status that can be displayed. The columns in the table are:

Status The node status that is displayed.

STATUS parameter
The STATUS() filter that will select nodes with the specified status.

Scope The scope of the status. GBL indicates that the status can be global (it exists in the resource structure when STM is enabled), and is returned with the STT short label. LCL indicates that the status can be local, and is returned with the LSTT short label.

Meaning
Provides a brief description of the status.

Table 74. QUERY NODE status

Status	STATUS parameter	Scope	Meaning
ACTIVE	ACTIVE	LCL	Node is in an XRF session on the active system.
ASR	ASR	LCL	Node has session initiation option ASR.
BACKUP	BACKUP	LCL	Node is in an XRF session on the alternate system.
CLSDST	CLSDST	LCL	Session is being disconnected.
COLD	COLD	GBL and LCL	For a SLUP or FINANCE terminal, indicates that the next session initiation is cold (message sequence numbers are initialized to 0).
CON	CON	LCL	Node is connected or in session with IMS.
CONVACT	CONV	GBL and LCL	An active conversation exists.
CONVHELD	CONV	GBL and LCL	One or more held conversations exist.
C1INOP C2INOP C3INOP C4INOP	CINOP	LCL	Indicates the inoperable node or terminal component, where C1, C2, C3 and C4 refer to the component as defined by system definition. (See the /COMPT command and the /RCOMPT command for details on how to ready inoperable components).
DEACT	DEACT	LCL	Node has been permanently deactivated. Restart of node requires /STOP DC and /START DC commands. Alternatively, node can be activated with UPDATE NODE STOP(DEACT). Message DFS2473 in the system console log might contain information regarding the reason this status was set. DFS2473 can occur more than once in the system console log.
EXCL	EXCL	GBL and LCL	The node is in exclusive mode set by the /EXCLUSIVE command. Exclusive mode restricts the output received by the terminal affected.
FORCSESS	FORCSESS	LCL	Node has session initiation option of FORCE.

Table 74. QUERY NODE status (continued)

Status	STATUS parameter	Scope	Meaning
IDLE	IDLE	LCL	No activity of any kind is in progress for the node.
INOP	INOP	LCL	NODE is inoperable.
LOCK	LOCK	LCL	Node is locked, set by the /LOCK command. The sending and receiving of messages for the VTAM node is stopped.
LOST	LOST	LCL	The VTAM LOSTERM EXIT has been scheduled for this node but has not yet been recognized by IMS. At the next interrupt for this node, IMS interrogates the LOSTERM value. All values, with one exception, result in an immediate CLSDST, or disconnection from IMS. For the LOSTERM exception, IMS must wait for VTAM to notify IMS (via another LOSTERM) of completion of recovery operation. This indicates an IMS logger's connection to the Transport Manager Subsystem is gone because of TMS or VTAM failure.
MFST	MFST	GBL and LCL	Node is in MFSTEST mode, set by UPDATE NODE START(MFST) command or /TEST MFS command. Terminals supported by Message Format Service use format blocks from a special test library if the requested format block is in the test library; otherwise the blocks are obtained from the production library.
OPNDST	OPNDST	LCL	OPNDST is in progress for this node.
PAGE	PAGE	LCL	Indicates an MFS paged message.
PRI	PRI	GBL and LCL	This node is the primary partner of an ISC session.
PRST	PRST	LCL	The node is in preset destination mode. PRST mode is established by the /SET command. All messages entered at this terminal are sent to the preset destination transaction code or logical terminal.
QUI	QUI	LCL	A VTAM node has sent a VTAM Quiesce-End-of-Chain indicator to suspend IMS output.
RELREQ	RELREQ	LCL	VTAM RELREQ exit routine has been driven but IMS is waiting for an operation in progress to complete before releasing the node.
RESP	RESP	GBL and LCL	The node is in response mode and the response reply message is available for output or in the process of being sent.
RESPINP	RESPINP	GBL and LCL	The node is in response mode and the response mode input is still in-doubt; for example, the response reply message is not available for output.

Table 74. QUERY NODE status (continued)

Status	STATUS parameter	Scope	Meaning
RESPINPFP	RESPINPFP	GBL and LCL	The node is in Fast Path response mode and the response mode input is still in-doubt; for example, the response reply message is not available for output.
RESYNC	RESYNC	LCL	The positive acknowledgment of an IMS recoverable output message was not received when the connection with the VTAM node was terminated. This message is subject to resynchronization when the next connection for this node is attempted.
RM	RM	GBL	The node exists in the resource structure managed by RM.
RMACTIVE	RMACTIVE	GBL	The node is active (logged-on) in the IMSplex, as indicated in the RM structure (RM active).
RMOWNED	RMOWNED	GBL	The node is owned by an IMS system in the IMSplex, as indicated in the RM structure (RM owned).
SEC	SEC	GBL and LCL	This node is the secondary partner of an ISC session.
SHUT	SHUT	LCL	Normal processing has completed for the node and a VTAM shutdown-complete indicator was returned to IMS. The node can receive IMS output but cannot enter data while in this state.
SIGN	SIGN	GBL and LCL	The user is signed on to a node under enhanced security.
SIMLOGON	SIMLOGON	LCL	A logon to IMS has been simulated.
STATIC	STATIC	GBL and LCL	The node was defined during system definition.
STOLGN	STOLGN	GBL and LCL	Node was stopped from logging on by the UPDATE NODE STOP(LOGON) command.
TEST	TEST	LCL	The node is in test mode.
TKOTRA	TKOTRA	LCL	Node in an XRF session is to be traced only during takeover, to help diagnose XRF terminal switch problems.
TRACE	TRACE	GBL and LCL	Node is being traced.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY NODE command. The OM return and reason codes that may be returned as a result of the QUERY NODE command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 75. Return and reason codes for the QUERY NODE command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001000'	The command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000008'	X'00002014'	An invalid character was specified in the resource name.
X'00000008'	X'00002040'	An invalid parameter value was specified. An invalid SHOW or STATUS value might have been specified.
X'00000008'	X'00002133'	The NAME() keyword specified is either a generic name or multiple specific names, but the USER() keyword was also specified. When USER() is specified, only one NAME parameter is allowed.
X'0000000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See Table 76 on page 175 for details.
X'0000000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code. See Table 76 on page 175 for details.
X'00000010'	X'00004004'	Command processing terminated because CQS was not active.
X'00000010'	X'00004005'	Command processing terminated because CQS was not connected to the queue structure.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	Command processing terminated because the resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'00004028'	Global queue counts were requested, but shared message queues are not enabled.
X'00000010'	X'0000403C'	VTAM generic resource affinity information was requested, but VGR is not enabled.
X'00000010'	X'00004104'	Command processing terminated because RM is not available.
X'00000010'	X'00004108'	Command processing terminated because SCI is not available.
X'00000014'	X'00005004'	A DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	Command processing terminated because of an RM error.
X'00000014'	X'00005104'	Command processing terminated because of a CQS error.
X'00000014'	X'00005108'	Command processing terminated because of an SCI error.
X'00000014'	X'00005FFF'	Command processing terminated because of an internal IMS error.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 76. Completion codes for the QUERY NODE command

Completion code	Completion code text	Meaning
0		The QUERY NODE command completed successfully for the resource.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
98	CQS REQUEST ERROR	Global queue counts could not be obtained because of a CQS error.
1A1	Node resource is in error	The node resource was found in the resource structure and an associated resource was needed, but it was either not found or appeared to be in error. This is normally an error condition. However, it could be a temporary condition caused by terminal or command activity. The command should be retried.

Examples

The following are examples of the QUERY NODE command:

Example 1 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE2*,XYZ) SHOW(LOCAL)
```

TSO SPOC output:

```
Node     ISCUser  MbrName CC CCText
NODE21            IMS1     0
NODE22            IMS1     0
NODE22            IMS2     0
NODE23            IMS2     0
NODE24   USER24A  IMS2     0
NODE24   USER24B  IMS2     0
NODE24   N/A      IMS2     0
XYZ               IMS1    10 NO RESOURCES FOUND
XYZ               IMS2    10 NO RESOURCES FOUND
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM and shared queues are irrelevant because only local information is requested. IMS1, the command master, displays only local information because no global information is requested. IMS2 displays local information only.

- NODE21 exists on IMS1 only.

- NODE22 exists on IMS1 and IMS2.
- NODE23 exists on IMS2 only.
- NODE24 is an ISC node with 3 parallel sessions available on IMS2, two of which are allocated.
- XYZ does not exist on any system.

Example 2 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE2*)
```

TSO SPOC output:

```
Node     ISCUser  MbrName CC  Gbl
NODE21            IMS1     0  Y
NODE21            IMS1     0
NODE22            IMS1     0
NODE22            IMS2     0
NODE23            IMS1     0  Y
NODE23            IMS2     0
NODE24   USER24A  IMS1     0  Y
NODE24   USER24B  IMS1     0  Y
NODE24   N/A      IMS1     0  Y
NODE24   USER24A  IMS2     0
NODE24   USER24B  IMS2     0
NODE24   N/A      IMS2     0
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information only.

- NODE21 exists on IMS1 and in the resource structure.
- NODE22 exists on IMS1 and IMS2 only.
- NODE23 exists on IMS2 and in the resource structure.
- NODE24 is an ISC node with 3 parallel sessions available on IMS2, two of which are allocated on IMS2 and in the resource structure. IMS1 also displays an output line for NODE24 with N/A as the ISC user, which represents any global status that the node might have (that is not tied to any particular parallel session).

Example 3 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE2*) STATUS(STATIC) SHOW(LOCAL)
```

TSO SPOC output:

```
Node     ISCUser  MbrName CC  LclStat
NODE21            IMS1     0  IDLE,CONVACT,CON,STATIC
NODE22            IMS1     0  IDLE,STATIC
NODE22            IMS2     0  IDLE,STATIC
NODE24   USER24A  IMS2     0  IDLE,CON,PRI,STATIC
NODE24   USER24B  IMS2     0  IDLE,CON,PRI,STATIC
NODE24   N/A      IMS2     0  IDLE,STATIC
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays local information only

because SHOW(LOCAL) is specified. IMS2 displays local information only. All static nodes are displayed, and status is displayed, because the STATUS filter was specified.

- NODE21 exists on IMS1, is logged on, and has an active conversation.
- NODE22 exists on IMS1 and IMS2.
- NODE24 is an ISC node with 3 parallel sessions available on IMS2, two of which are allocated and are the primary partners.

Example 4 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE21) SHOW(GLOBAL,CONV,LTERM,STATUS)
```

TSO SPOC output:

```
(screen 1)
Node     MbrName CC  Gbl  Lterm    ConvID ConvTran ConvStat
NODE21   IMS1     0  Y
NODE21   IMS1     0  Y    LTERM21A
NODE21   IMS1     0  Y    LTERM21B
NODE21   IMS1     0  Y                  1 TRAN1A   CONVHELD
NODE21   IMS1     0  Y                  2 TRAN1A   CONVHELD
NODE21   IMS1     0  Y                  3 TRAN1A   CONVACTV

(scrolled right to screen 2)
Node     MbrName Gbl  Status
NODE21   IMS1    Y    CONVACT,STATIC,RM,RMACTIVE,RMOWNED
NODE21   IMS1    Y
NODE21   IMS1    Y
NODE21   IMS1    Y
NODE21   IMS1    Y
NODE21   IMS1    Y
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global information only. IMS2 ignores the command (RC=4, RSN=x1000) because only global information is requested.

NODE21 exists in the resource structure. IMS1 displays a global line which shows that the node is active in the IMSplex, and has a conversation active. There are two logical terminals assigned to the node, and are displayed on separate output lines. There are three conversations associated with the node, and are displayed on separate output lines.

Example 5 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE23) SHOW(CONV,STATUS,OWNER,RECOVERY)
```

TSO SPOC output:

```
(screen 1)
Node     MbrName CC CCText             Gbl  Owner  SRM  Rcvy
NODE23   IMS1     0                    Y    IMS2   LCL  CONV,FP
NODE23   IMS1    10 NO RESOURCES FOUND
NODE23   IMS2     0
NODE23   IMS2     0
NODE23   IMS2     0

(scrolled right to screen 2)
```

```
Node     MbrName Gbl  ConvID ConvTran ConvStat Status
NODE23   IMS1    Y                             RM,RMACTIVE,RMOWNED
NODE23   IMS1
NODE23   IMS2
NODE23   IMS2
NODE23   IMS2

(scrolled right to screen 3)
Node     MbrName Gbl  LSRM LRcvy    LConvID LConvTran LConvStat
NODE23   IMS1    Y
NODE23   IMS1
NODE23   IMS2         LCL  CONV,FP
NODE23   IMS2                           1 TRAN1A    CONVHELD
NODE23   IMS2                           2 TRAN1B    CONVACTV

(scrolled right to screen 4)
Node     MbrName Gbl  LclStat
NODE23   IMS1    Y
NODE23   IMS1
NODE23   IMS2         IDLE,CONVACT,CON
NODE23   IMS2
NODE23   IMS2
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information.

NODE23 exists in IMS2 and in the resource structure. IMS1 displays a global line which shows the node is active and owned on IMS2, and its status recovery mode is LOCAL, which means conversation information is not known globally. IMS1 also displays a local line showing that NODE23 does not exist locally. IMS2 displays the local information, which includes one status line, and an additional output line for each conversation active or held locally.

Example 6 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE23) SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
Node     MbrName CC CCText             Gbl QCnt EMHQCnt Type  Owner  SRM  Rcvy
NODE23   IMS1     0                    Y      0       0 SLU2  IMS2   LCL  CONV,FP
NODE23   IMS1    10 NO RESOURCES FOUND
NODE23   IMS2     0

(scrolled right to screen 2)
Node     MbrName Gbl  User     Userid   Affin    Version# Version#SNU Status
NODE23   IMS1    Y    USER23   UID23    IMS2            5           0 RM,RMACTIVE,RMOWNED
NODE23   IMS1
NODE23   IMS2

(scrolled right to screen 3)
Node     MbrName Gbl  LQCnt LType CID      RecdCnt SentCnt DefMdtbl ActMdtbl
NODE23   IMS1    Y
NODE23   IMS1
NODE23   IMS2             0 SLU2  02000003       9      13 SLU2MOD2 SLU2MOD2

(scrolled right to screen 4)
Node     MbrName Gbl  LSRM Lrcvy    LUser    LUserid  LVersion# LVersion#SNU LclStat
```

```
NODE23   IMS1    Y
NODE23   IMS1
NODE23   IMS2        LCL  CONV,FP  USER23   UID23           5           0 IDLE,CONVACT,CON
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are active. IMS1, the command master, displays global and local information. IMS2 displays local information.

NODE23 exists in IMS2 and in the resource structure. IMS1 displays a global line which shows global queue counts and global status from the resource structure. Global status indicates that the node is active on IMS2, has a VGR affinity with IMS2, and its status recovery mode is LOCAL, which means conversation information is not known globally. IMS1 also displays a local line showing that NODE23 does not exist locally. IMS2 displays the local information, which shows that conversation status exists locally.

Example 7 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(DFSLN001,DFSLN002) SHOW(LOCAL,TYPE)
```

TSO SPOC output:

```
Node     Line Pterm MbrName CC LType
DFSLN001    1     1 IMS1     0 CONSOLE
DFSLN001    1     1 IMS2     0 CONSOLE
DFSLN002    2     1 IMS1     0 RDR/PTR
DFSLN002    2     1 IMS2     0 RDR/PTR
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM and shared queues are irrelevant because only local information is requested. IMS1, the command master, displays only local information because no global information is requested. IMS2 displays local information only.

DFSLN001 represents the system console, LINE 1 PTERM 1, on each system. DFSLN002 represents a SYSOUT or SPOOL device, LINE 2 PTERM 1, on each system.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY ASMT command (Commands)

/DISPLAY CONV command (Commands)

/DISPLAY NODE command (Commands)

/DISPLAY STATUS command (Commands)

/DISPLAY TRACE command (Commands)

Command keywords and their synonyms (Commands)

QUERY ODBM commands

Use the QUERY ODBM commands to query information about Open Database Manager (ODBM), a component of the Common Service Layer (CSL).

Subsections:

- "QUERY ODBM TYPE(ALIAS) command"
- "QUERY ODBM TYPE(CONFIG) command" on page 183
- "QUERY ODBM TYPE(DATASTORE) command" on page 187
- "QUERY ODBM TYPE(SCIMEMBER) command" on page 191
- "QUERY ODBM TYPE(THREAD) command" on page 194
- "QUERY ODBM TYPE(TRACE) command" on page 202

QUERY ODBM TYPE(ALIAS) command

Use the QUERY ODBM TYPE(ALIAS) command to query information about Open Database Manager (ODBM) configuration.

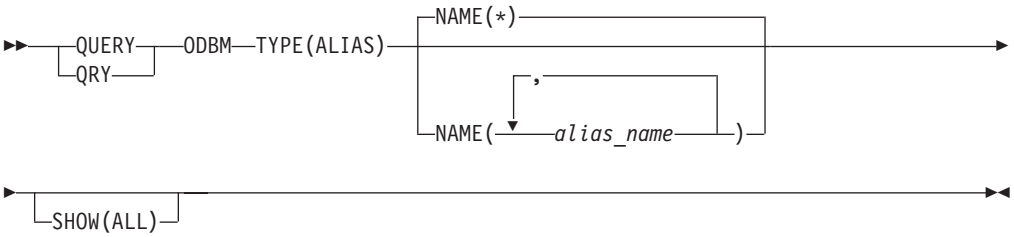
Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 181
- "Output fields" on page 181
- "Return, reason, and completion codes" on page 181
- "Examples" on page 182

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the QUERY ODBM TYPE(ALIAS) command.

NAME()
Specifies the alias name of the IMS data store to be queried, as defined in the ALIAS substatement of the ODBM configuration statement in the CSLDCxxx configuration member. Wildcards (* and %) can be specified for *alias_name*. The

alias_name is a repeatable parameter. The default is NAME(*), which returns information about all aliases that are known to ODBM.

SHOW()
Specifies the ODBM TYPE(ALIAS) output fields to be returned. The parameters that are supported with the SHOW keyword are:

ALL
Returns all information about ODBM data store alias and its status.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 77. Output field descriptions for the QUERY ODBM TYPE(ALIAS) command

Short label	Long label	Keyword	Meaning
ALCSTT	AliasStatus	N/A	Alias connection status. Status can be either STARTED or STOPPED.
ALIAS	AliasName	N/A	Alias name.
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
DSCSTT	DatastoreStatus	N/A	Connection status of the data store associated with the alias.
DSTR	DatastoreName	N/A	Data store name for the alias.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
THDCT	ThreadCount	N/A	Count of threads that are associated with the alias.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(ALIAS) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 78. Return and reason codes for the QUERY ODBM TYPE(ALIAS) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(ALIAS) command completed successfully.
X'40000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'40000008'	X'00002008'	Insufficient number of keywords.
X'40000008'	X'00002014'	Invalid character in resource name.
X'40000008'	X'0000203C'	Invalid parameter specified.
X'4000000C'	X'00003000'	At least one request was successful.
X'4000000C'	X'00003004'	No requests were successful.
X'40000014'	X'00005034'	An OM response buffer request failed.
X'40000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 79. Completion codes for the QUERY ODBM TYPE(ALIAS) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(ALIAS) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
2	PARAMETER SPECIFIED TOO LONG	The NAME() parameter specified is longer than the valid alias name length of 4 characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(ALIAS) command

TSO SPOC input:

```
QRY ODBM TYPE(ALIAS) NAME(*) SHOW(ALL)
```

TSO SPOC output:

```
Mbrname     AliasName    CC   AliasStatus  ThreadCount  DatastoreName  DatastoreStatus
ODBM01OD      IO1A        0    STARTED           5         IMS1           STARTED
ODBM01OD      IO1B        0    STOPPED           0         IMS1           STARTED
ODBM01OD      IO2A        0    STARTED          10         IMS2           STARTED
ODBM01OD      IO2B        0    STARTED           0         IMS2           STARTED
ODBM02OD      IO3A        0    STOPPED           0         IMS3           STOPPED
ODBM02OD      IO3B        0    STARTED           0         IMS3           STOPPED
```

Explanation: The QUERY command displays alias information for all aliases that are known to ODBM.

QUERY ODBM TYPE(CONFIG) command

Use the QUERY ODBM TYPE(CONFIG) command to query information about Open Database Manager (ODBM) configuration.

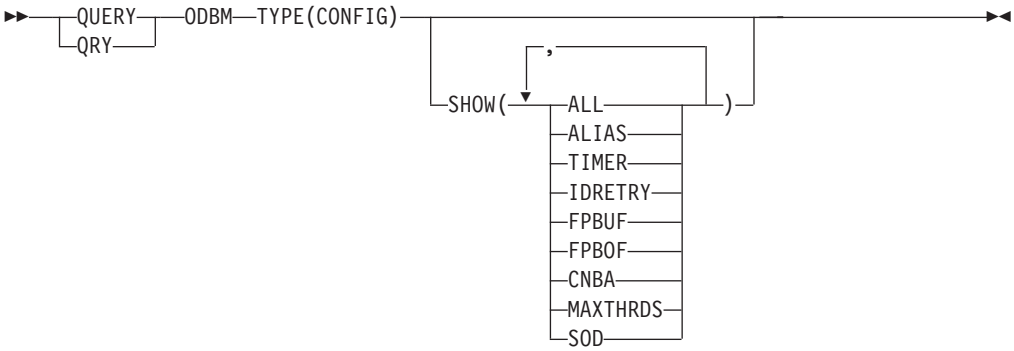
Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 184
- "Output fields" on page 184
- "Return, reason, and completion codes" on page 185
- "Examples" on page 186

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the QUERY ODBM TYPE(CONFIG) command.

SHOW()
Specifies the ODBM TYPE(CONFIG) output fields to be returned. The parameters supported with the SHOW keyword are:

ALL
Returns all information about the CSLDCxxx configuration member in PROCLIB.

ALIAS
Returns alias names for the data store, which are used by Open Database applications to access IMS data stores.

TIMER
Returns the amount of time (in seconds) between attempts by ODBM to establish a connection to an IMS data store. The value can be a number between 1–99.

IDRETRY
Returns the number of times ODBM will attempt to connect (identify) to an IMS data store after the initial attempt is unsuccessful. The value can be a number between 0–255.

FPBUF
Returns the number of Fast Path DEDB buffers to be allocated and fixed per thread. The value can be a number between 00–999.

FPBOF
Returns the number of Fast Path DEDB overflow buffers to be allocated per thread. The value can be a number between 00– 999.

CNBA
Returns the total number of Fast Path NBA buffers for ODBM use. The value can be a number between 0– 9999.

MAXTHRDS
Returns the maximum number of concurrent active threads to an individual IMS data store. The value can be a number between 1–999.

SOD
Returns the SYSOUT class of the SNAP dumps produced by the ODBM address space. The value returned is a single alphanumeric character.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 80. Output field descriptions for the QUERY ODBM TYPE(CONFIG) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.

Table 80. Output field descriptions for the QUERY ODBM TYPE(CONFIG) command (continued)

Short label	Long label	Keyword	Meaning
CNBA	CNBA	CNBA	CNBA value for a data store connection.
CNFG	ConfigName	N/A	CSLDCxxx PROCLIB member name.
DSTR	DatastoreName	N/A	Data store name.
FPBOF	FPBOF	FPBOF	FPBOF value for a data store connection.
FPBUF	FPBUF	FPBUF	FPBUF value for a data store connection.
GBL	Global	N/A	Y or N value will be presented. Y indicates information is from the Global section in the CSLDCxxx configuration member. N indicates information is from the Local section in the CSLDCxxx configuration member.
IRTRY	IdRetry	IDRETRY	IDRETRY value of a data store connection.
MALIAS	Aliases	ALIAS	The alias names associated with an IMS data store connection.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
MXTHD	MaxThrds	MAXTHRDS	MAXTHRD parameter value.
SOD	SOD	SOD	SYSOUT class of ODBM snap dumps.
TIMER	Timer	TIMER	TIMER parameter value for a data store connection.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(CONFIG) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 81. Return and reason codes for the QUERY ODBM TYPE(CONFIG) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(CONFIG) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.

Table 81. Return and reason codes for the QUERY ODBM TYPE(CONFIG) command (continued)

Return code	Reason code	Meaning
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 82. Completion codes for the QUERY ODBM TYPE(CONFIG) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(CONFIG) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
10	NO RESOURCES	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(CONFIG) command

TSO SPOC input:

```
QRY ODBM TYPE(CONFIG) SHOW(ALL)
```

TSO SPOC output:

```
MbrName    ConfigName  Global DatastoreName  CC  FPBUF FPBOF CNBA MaxThrds
ODBM01OD    CSLDC000     Y                    0     5     5  100       1
ODBM01OD    CSLDC000     N     IMS1           0     0     0    0      10
ODBM01OD    CSLDC000     N     IMS2           0    50    50  500       5
ODBM02OD    CSLDC000     Y                    0     5     5  200      10
ODBM02OD    CSLDC000     N     IMS3           0    10    10  300       5

(continued)
IDRETRY   TIMER    Aliases     SOD
       5     30                  I
                   IO1A,IO1B     E
                   IO2A,IO2B     E
      10     20                  I
                   IO3A,IO3B     E
```

Explanation: The QUERY command will display all SHOW parameter values for the CSLDCxxx configuration members.

Example 2 for QUERY ODBM TYPE(CONFIG) command

TSO SPOC input:

```
QRY ODBM TYPE(CONFIG) SHOW(ALIAS,CNBA)
```

TSO SPOC output:

```
MbrName    ConfigName Global DatastoreName    CC     CNBA     Aliases
ODBM01OD    CSLDC000   Y                      0        0
ODBM01OD    CSLDC000   N      IMS1            0        0     IO1A,IO1B
```

```
ODBM01OD    CSLDC000   N      IMS2            0      500     IO2A,IO2B
ODBM02OD    CSLDC000   Y                      0      100
ODBM02OD    CSLDC000   N      IMS3            0      200     IO3A,IO3B
```

Explanation: The QUERY command will display information about alias names for the data store and the total number of Fast Path NBA buffers for ODBM use.

Example 3 for QUERY ODBM TYPE(CONFIG) command

TSO SPOC input:

```
QRY ODBM TYPE(CONFIG)
```

TSO SPOC output:

```
MbrName     ConfigName Global DatastoreName  CC
ODBM01OD    CSLDC000   Y                      0
ODBM01OD    CSLDC000   N        IMS1          0
ODBM01OD    CSLDC000   N        IMS2          0
ODBM01OD    CSLDC000   N        IMS1          0
ODBM01OD    CSLDC000   N        IMS1          0
ODBM02OD    CSLDC000   Y                      0
ODBM02OD    CSLDC000   N        IMS3          0
```

Explanation: The QUERY command will display information about Open Database Manager (ODBM) configuration.

QUERY ODBM TYPE(DATASTORE) command

Use the QUERY ODBM TYPE(DATASTORE) command to query information about an IMS data store and its associated resources.

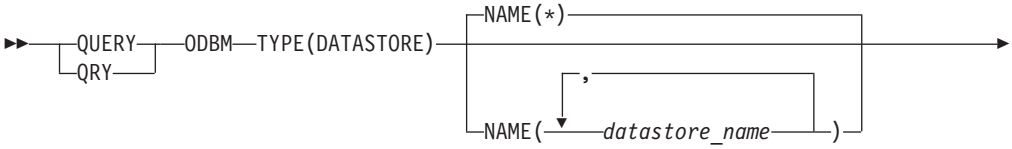
Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 188
- "Usage notes" on page 189
- "Output fields" on page 189
- "Return, reason, and completion codes" on page 190
- "Examples" on page 190

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the QUERY ODBM TYPE(DATASTORE) command.

NAME()
Specifies the name of the data store to be displayed. Wildcards (* and %) can be specified for *datastore_name*. The *datastore_name* is a repeatable parameter. The default is NAME(*), which returns information about all data stores that are known to ODBM.

SHOW()
Specifies the ODBM TYPE(DATASTORE) output fields to be returned. One or more parameters can be specified. The parameters supported with the SHOW keyword are:

ALL
Returns all information about the specified IMS data store.

ALIAS
Returns alias names for the IMS data store. Client application programs identify the IMS data store that they are accessing by the alias name.

TIMER
Returns the amount of time (in seconds) between attempts by ODBM to establish a connection to an IMS data store. The value can be a number between 1–99.

IDRETRY
Returns the number of times ODBM will attempt to connect (identify) to an IMS data store after the initial attempt is unsuccessful. The value can be a number between 0–255.

FPBUF
Returns the number of Fast Path DEDB buffers to be allocated and fixed per thread. The value can be a number between 00–999.

FPBOF
Returns the number of Fast Path DEDB overflow buffers to be allocated per thread. The value can be a number between 00– 999.

CNBA
Returns the total number of Fast Path NBA buffers for ODBM use. CNBA pertains to each ODBM-to-IMS connection. The value can be a number between 0– 9999.

MAXTHRDS
Returns the maximum number of concurrent active threads to an individual IMS data store. The value can be a number between 1–999.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 83. Output field descriptions for the QUERY ODBM TYPE(DATASTORE) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
CNBA	CNBA	CNBA	CNBA value in use for the data store connection.
DSCSTT	ConnectionStatus	N/A	Data store connection status.
DSTR	DatastoreName	N/A	Data store name.
FPBOF	FPBOF	FPBOF	FPBOF value for the data store connection.
FPBUF	FPBUF	FPBUF	FPBUF value for the data store connection.
IRTRY	IdRetry	IDRETRY	IDRETRY value of the data store connection.
MALIAS	Aliases	ALIAS	The alias names associated with the IMS data store connection.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
MXTHD	MaxThrds	MAXTHRDS	MAXTHRD parameter value.
THDCT	ThreadCount	N/A	Count of threads associated to the data store connection.
TIMER	Timer	TIMER	TIMER parameter value for the data store connection.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(DATASTORE) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 84. Return and reason codes for the QUERY ODBM TYPE(DATASTORE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(DATASTORE) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 85. Completion codes for the QUERY ODBM TYPE(DATASTORE) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(DATASTORE) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
2	PARAMETER SPECIFIED TOO LONG	The NAME() parameter specified is longer than the valid data store name length of 8 characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(DATASTORE) command

TSO SPOC input:

```
QRY ODBM TYPE(DATASTORE) NAME(IMS*) SHOW(ALL)
```

TSO SPOC output:

```
MbrName    DatastoreName  CC  ConnectionStatus  ThreadCount  FPBUF  FPBOF  CNBA
ODBM01OD    IMS1           0      STARTED                5      0      0     0
ODBM01OD    IMS2           0      STOPPED                0     50     50   500
ODBM02OD    IMS3           0      STARTED               10     10     10   200

(continued)

MaxThrds IDRETRY TIMER  Aliases
      10       5    30  IO1A,IO1B
       5       5    30  IO2A,IO2B
      10       5    30  IO3A,IO3B
```

Explanation: The QUERY command will display all information about the data stores whose name begin with IMS.

Example 2 for QUERY ODBM TYPE(DATASTORE) command

TSO SPOC input:

```
QRY ODBM TYPE(DATASTORE) NAME(IMS*) SHOW(ALIAS,MAXTHRDS)
```

TSO SPOC output:

```
MbrName    DatastoreName  CC  ConnectionStatus  ThreadCount  MaxThrds  Aliases
ODBM01OD    IMS1           0  STARTED                    5        10   IO1A,IO1B
ODBM01OD    IMS2           0  STOPPED                    0         5   IO2A,IO2B
ODBM02OD    IMS3           0  STARTED                   10        10   IO3A,IO3B
```

Explanation: The QUERY command will display information about alias names for the data store and the maximum number of concurrent active threads to an individual IMS data store.

Example 3 for QUERY ODBM TYPE(DATASTORE) command

TSO SPOC input:

```
QRY ODBM TYPE(DATASTORE) NAME(IMS*)
```

TSO SPOC output:

```
MbrName    DatastoreName  CC  ConnectionStatus  ThreadCount   Aliases
ODBM01OD    IMS1           0        STARTED              5    IO1A,IO1B
ODBM01OD    IMS2           0        STOPPED              0    IO2A,IO2B
ODBM02OD    IMS3           0        STARTED             10    IO3A,IO3B
```

Explanation: The QUERY command will display information about data store connection status, thread count, and alias names for the data store.

QUERY ODBM TYPE(SCIMEMBER) command

Use the QUERY ODBM TYPE(SCIMEMBER) command to query information about an Open Database Manager (ODBM) client.

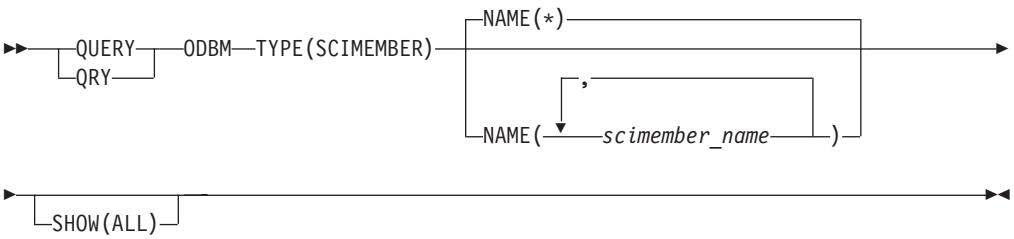
Subsections:

- "Environment" on page 192
- "Syntax" on page 192
- "Keywords" on page 192
- "Usage notes" on page 192
- "Output fields" on page 192
- "Return, reason, and completion codes" on page 193
- "Examples" on page 194

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the QUERY ODBM TYPE(SCIMEMBER) command.

NAME()
Specifies the name of the ODBM client to be displayed. Wildcards (* and %) can be specified for *scimember_name*. The *scimember_name* is a repeatable parameter. The default is NAME(*), which returns information about all SCI member names that are known to ODBM.

SHOW()
Specifies the ODBM TYPE(SCIMEMBER) output fields to be returned. One or more parameters can be specified. The parameters supported with the SHOW keyword are:

ALL
Returns all information about the ODBM clients that have the specified SCI member names.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning

Provides a brief description of the output field.

Table 86. Output field descriptions for the QUERY ODBM TYPE(SCIMEMBER) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
SCMBR	SCIMbrName	N/A	The SCI member name of the client that is connected to ODBM.
THDCT	ThreadCount	N/A	Count of threads that are associated with the SCI member.
TYPE	Type	N/A	SCI member type of the client that is connected to ODBM.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(SCIMEMBER) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 87. Return and reason codes for the QUERY ODBM TYPE(SCIMEMBER) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(SCIMEMBER) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 88. Completion codes for the QUERY ODBM TYPE(SCIMEMBER) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(SCIMEMBER) command completed successfully.

Table 88. Completion codes for the QUERY ODBM TYPE(SCIMEMBER) command (continued)

Completion code	Completion code text	Meaning
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
2	PARAMETER SPECIFIED TOO LONG	The NAME() parameter specified is longer than the valid ODBM client name length of 8 characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(SCIMEMBER) command

TSO SPOC input:

```
QRY ODBM TYPE(SCIMEMBER) NAME(HWS*) SHOW(ALL)
```

TSO SPOC output:

```
MbrName       SCIMbrName  CC   Type     ThreadCount
ODBM01OD         HWS1      0   IMSCON         21
ODBM01OD         HWS2      0   IMSCON          5
```

Explanation: The QUERY command will display ODBM client information about SCI members whose names begin with HWS.

QUERY ODBM TYPE(THREAD) command

Use the QUERY ODBM TYPE(THREAD) command to query information about an Open Database Manager (ODBM) thread and its associated resources.

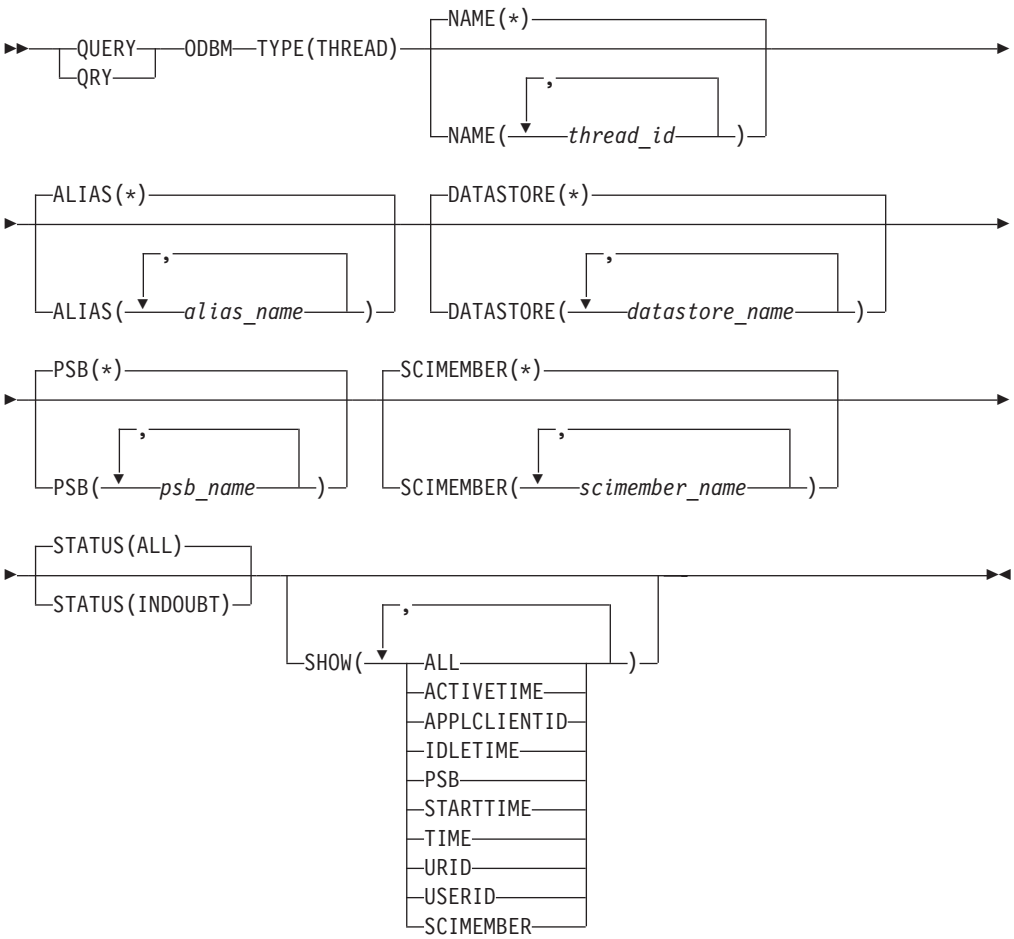
Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 195
- "Usage notes" on page 197
- "Output fields" on page 197
- "Return, reason, and completion codes" on page 198
- "Examples" on page 199

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the QUERY ODBM TYPE(THREAD) command.

ALIAS(*alias_name***)**
Specifies the 1–4 character alias name of the data store to be used as a filter. Thread information is returned if the *thread_id* is associated with the specified *alias_name*.

Wildcards (* and %) can be specified for the *alias_name*. The *alias_name* is a repeatable parameter. The default is ALIAS(*), which returns thread information for the specified *thread_id* regardless of alias affiliation.

DATASTORE(*datastore_name***)**
Specifies the 1–8 character name of the data store to be used as a filter. Thread information is returned if the specified *thread_id* is associated with the *datastore_name* specified.

Wildcards can be specified for the *datastore_name*. The *datastore_name* is a repeatable parameter. The default is DATASTORE(*), which returns thread information for the specified *thread_id* regardless of data store affiliation.

NAME(*thread_id***)**
Specifies the thread ID of the ODBM thread you want to be displayed. Wildcards (* and %) can be specified for the *thread_id*. The *thread_id* is a

repeatable parameter. The default is NAME(*), which returns information about all threads that are known to ODBM. The length of each thread ID can be up to 32 characters.

PSB(*psb_name***)**
Specifies the 1–8 character name of the PSB to be used as a filter. Thread information is returned if the specified *thread_id* is associated with the *psb_name* specified.

Wildcards (* and %) can be specified for the *psb_name*. The *psb_name* is a repeatable parameter. The default is PSB(*), which returns thread information for the specified *thread_id* regardless of PSB affiliation.

Note: If a PSB name is specified for the QRY ODBM TYPE(THREAD) command, PSB names will be included in the command output even if SHOW(PSB) is not specified.

SCIMEMBER(*scimember_name***)**
Specifies the 1–8 character name of the client connected to ODBM to be used as a filter. Thread information is returned if the *thread_id* is associated with the *scimember_name* specified.

Wildcards (* and %) can be specified for the *scimember_name*. The *scimember_name* is a repeatable parameter. The default is SCIMEMBER(*), which returns thread information for the *thread_id* regardless of ODBM client affiliation.

Note: If the SCIMEMBER keyword is specified for the QRY ODBM TYPE(THREAD) command, ODBM client names will be returned in the command output even if SHOW(SCIMEMBER) is not specified.

STATUS()
Allows for filtering on the QUERY ODBM TYPE(THREAD) command by the status of the thread. The following parameters can be specified:

ALL
Displays information about threads of any of the following status. STATUS(ALL) is the default.
- ACTIVE - The thread is active and is not in any of the following states.
- INPREPARE
- INCOMMIT
- INBACKOUT
- INDOUBT
- INTERM
- INABTERM

INDOUBT
Displays only threads with an INDOUBT status.

SHOW()
Specifies the ODBM TYPE(THREAD) output fields to be returned. One or more parameters can be specified. The parameters supported with the SHOW keyword are:

ALL
Returns all information about the ODBM thread.

ACTIVETIME
ACTIVETIME is defined as the time spent in a data store. Specifying SHOW(ACTIVETIME) returns the elapsed time from the time the data store is entered to the time that the QUERY command is issued. This value

is not cumulative; if processing is not in a data store when the command is issued, a zero value will be returned in the ACTIVETIME column.

The format of ACTIVETIME is:

```
HH:MM:SS.th
```

If ACTIVETIME exceeds 24 hours, the ACTIVETIME column will display:

```
24+ HOURS
```

APPLCLIENTID
Returns the ID of the application client that originated the IMS data store connection request.

IDLETIME
IDLETIME is defined as the time not in a data store. Specifying SHOW(IDLETIME) returns the elapsed time between returning from data store processing and the time that the QUERY command was issued. This value is not cumulative; if processing is in a data store when the command is issued, a zero value will be returned in the IDLETIME column.

The format of IDLETIME is:

```
HH:MM:SS.th
```

If IDLETIME exceeds 24 hours, the IDLETIME column will display:

```
24+ HOURS
```

PSB
Returns the name of the PSB being used by the thread.

SCIMEMBER
Returns the name of the ODBM client that is associated with the thread.

STARTTIME
Returns the time that the thread started. STARTTIME is UTC and is formatted as follows:

```
DDD-YYYY HH:MM:SS.thmiju
```

TIME
Returns the three time values: STARTTIME, IDLETIME, and ACTIVETIME.

URID
Returns the UR identifier that is associated with the thread.

USERID
Returns the user ID associated with the thread.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated.

error appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 89. Output field descriptions for the QUERY ODBM TYPE(THREAD) command

Short label	Long label	Keyword	Meaning
ACID	ApplClientID	APPLCLIENTID	The ID of the application client that originated the IMS data store connection request.
ATIME	ActiveTime	ACTIVETIME or TIME	Elapsed time that the thread has been processing in a data store.
ALIAS	AliasName	N/A	The alias name associated with the thread.
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
DSTR	DatastoreName	N/A	Data store name associated with the thread.
ITIME	IdleTime	IDLETIME or TIME	Elapsed time that the thread has been idle (Not processing in a data store).
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
PSB	PSB	PSB	Name of the PSB scheduled by the thread.
SCMBR	SCIMbrName	SCIMEMBER	The SCI member name of the client that is connected to ODBM for the thread.
STIME	StartTime	STARTTIME or TIME	Start time of the thread.
STT	Status	N/A	Status of the thread.
THID	ThreadID	N/A	Thread identifier.
URID	URID	URID	Unit of recovery identifier for the thread.
UID	Userid	USERID	User name associated with the thread.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(THREAD) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 90. Return and reason codes for the QUERY ODBM TYPE(THREAD) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(THREAD) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.

Table 90. Return and reason codes for the QUERY ODBM TYPE(THREAD) command (continued)

Return code	Reason code	Meaning
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'0000501C'	APSB hash table SCAN failed.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 91. Completion codes for the QUERY ODBM TYPE(THREAD) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(THREAD) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
2	PARAMETER SPECIFIED TOO LONG	The NAME() parameter specified is longer than the valid thread name length of 32 characters.
7	NO MATCHES FOUND FOR FILTER	The resource was found, but one or more of the specified filters did not match.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(*) SHOW(ALL)
```

TSO SPOC output:

```
TSO SPOC output:
MbrName   ThreadID                           CC    Status       DatastoreName
ODBM01OD  00000001233344129802334877222344  0     INBACKOUT       IMS1
ODBM01OD  00055001233344129802334877222981  0     INDOUBT         IMS2
ODBM01OD  01234444933344129802334877222344  0     INPREPARE       IMS3
ODBM02OD  92833457233344129802334877222344  0     INDOUBT         IMS4
ODBM03OD  63200001233344129802334877222344  0     INABTERM        IMS5

(continued)
```

```
AliasName  SCIMbrName  Userid    PSB    URID
IO1A       HWS1        User01    PSB1   0000000000000001
IO2B       HWS1        User02    PSB2   0000000000000025
IO3A       HWS2        User03    PSB3   0000000000000009
IO4A       HWS2        User03    PSB4   0000000000000087
IO5B       HWS1        User05    PSB5   0000000000000022

(continued)

StartTime                 IdleTime       ActiveTime   ApplClientID
182-2008 15:31:15.123456  00:00:00.01    00:00:00.00    CLIENT1
182-2008 15:31:29.237888  00:00:00.00    00:00:00.90    CLIENT2
182-2008 15:31:45.573900  00:00:00.02    00:00:00.00    CLIENT3
182-2008 15:31:18.000001  00:00:00.02    00:00:00.00    CLIENT3
182-2008 15:31:53.111242  00:00:00.01    00:00:00.00    CLIENT5
```

Explanation: The QUERY command will display ODBM thread information about all threads that are known to ODBM.

Example 2 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(*)
```

TSO SPOC output:

```
MbrName   ThreadID                           CC   Status       DatastoreName
ODBM01OD  00000001233344129802334877222344  0    INBACKOUT    IMS1
ODBM01OD  00055001233344129802334877222981  0    INDOUBT      IMS2
ODBM01OD  01234444933344129802334877222344  0    INPREPARE    IMS3
ODBM02OD  92833457233344129802334877222344  0    INDOUBT      IMS4
ODBM03OD  63200001233344129802334877222344  0    INABTERM     IMS5

(continued)

AliasName
IO1A
IO2B
IO3A
IO4A
IO5B
```

Explanation: The QUERY command will display information about all threads that are known to ODBM.

Example 3 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(*) DATASTORE(IMS1)
```

TSO SPOC output:

```
MbrName   ThreadID                           CC    Status     DatastoreName
ODBM01OD  00000001233344129802334877222344  0    INBACKOUT  IMS1

(continued)

AliasName
IO1A
```

Explanation: The QUERY command will display information about the threads that are associated with data store IMS1.

Example 4 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(*) SCIMEMBER(HWS2) SHOW(APPLCLIENTID,USERID)
```

TSO SPOC output:

```
MbrName    ThreadID                             CC  Status     DatastoreName
ODBM01OD   012344449333441298023348772223 44   0  INPREPARE  IMS3
ODBM02OD   928334572333441298023348772223 44   0  INDOUBT    IMS4

(continued)

AliasName  SCIMbrName  Userid    ApplClientID
IO3A          HWS2     User03    CLIENT3
IO4A          HWS2     User03    CLIENT3
```

Explanation: The QUERY command will display the application client identifier and the user ID of the threads that are associated with SCI member name HWS2.

Example 5 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(*) ALIAS(IO2*) SHOW(PSB,SCIMEMBER)
```

TSO SPOC output:

```
MbrName    ThreadID                             CC  Status    DatastoreName
ODBM01OD   000550012333441298023348772229 81   0  INDOUBT      IMS2


(continued)
AliasName      SCIMbrName  PSB
IO2B           HWS1        PSB2
```

Explanation: The QUERY command will display the PSB name and the ODBM client name for the thread whose alias begins with "IO2®".

Example 6 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(000*) PSB(P*) SHOW(USERID)
```

TSO SPOC output:

```
MbrName   ThreadID                           CC   DatastoreName    Status
ODBM01OD  00000001233344129802334877222344  0   IMS1             INBACKOUT
ODBM01OD  000550012333441298023348772229 81  0   IMS2             INDOUBT


(continued)

AliasName     Userid    PSB
IO1A          User01    PSB1
IO2B          User02    PSB2
```

Explanation: The QUERY command will display the user ID of the threads whose names begin with 000 and whose associated PSBs' names begin with "P".

Example 7 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(*) SHOW(TIME)
```

TSO SPOC output:

```
MbrName   ThreadID                           CC    Status       DatastoreName
ODBM01OD  00000001233344129802334877222344  0     INBACKOUT      IMS1
ODBM01OD  00055001233344129802334877222981  0     INDOUBT        IMS2
ODBM01OD  01234444933344129802334877222344  0     INPREPARE      IMS3
ODBM02OD  92833457233344129802334877222344  0     INDOUBT        IMS4
ODBM03OD  63200001233344129802334877222344  0     INABTERM       IMS5


(continued)

AliasName           StartTime                  IdleTime       ActiveTime
IO1A                182-2008 15:31:15.123456   00:00:00.01    00:00:00.00
IO2B                182-2008 15:31:29.237888   00:00:00.00    00:00:00.90
IO3A                182-2008 15:31:45.573900   00:00:00.02    00:00:00.00
IO4A                182-2008 15:31:18.000001   00:00:00.02    00:00:00.00
IO5B                182-2008 15:31:53.111242   00:00:00.01    00:00:00.00
```

Explanation: The QUERY command will display the starting time values of the threads that are associated with ODBM.

QUERY ODBM TYPE(TRACE) command

Use the QUERY ODBM TYPE(TRACE) command to query information about the Open Database Manager (ODBM) trace.

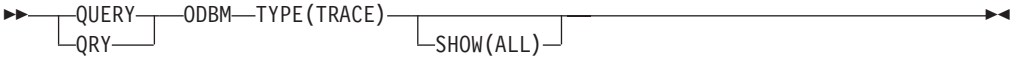
Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 203
- "Output fields" on page 203
- "Return, reason, and completion codes" on page 203
- "Examples" on page 204

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the QUERY ODBM TYPE(TRACE) command.

SHOW()
Specifies the ODBM TYPE(TRACE) output fields to be returned. The parameters supported with the SHOW keyword are:

ALL
Returns all information about the ODBM trace.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 92. Output field descriptions for the QUERY ODBM TYPE(TRACE) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
DSTR	DatastoreName	N/A	Data store name.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
TRACE	Trace	N/A	Trace information.
TRCSTT	TraceStatus	N/A	Trace status: Active or inactive.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(TRACE) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 93. Return and reason codes for the QUERY ODBM TYPE(TRACE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(TRACE) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.

Table 93. Return and reason codes for the QUERY ODBM TYPE(TRACE) command (continued)

Return code	Reason code	Meaning
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 94. Completion codes for the QUERY ODBM TYPE(TRACE) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(TRACE) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
10	NO RESOURCES	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(TRACE) command

TSO SPOC input:

```
QRY ODBM TYPE(TRACE) SHOW(ALL)
```

TSO SPOC output:

```
MbrName     Trace   DatastoreName  CC   TraceStatus
ODBM01OD    System  IMS1            0   Active
ODBM02OD    System  IMS3            0   Inactive
```

Explanation: The QUERY command will display all ODBM trace data.

QUERY OLC command

For an IMS running with RM (RMENV=Y), the QUERY OLC command returns information about the OLCSTAT DS that is shared by all of the IMS systems participating in global online change.

Subsections:

- "Environment" on page 205
- "Syntax" on page 205
- "Keywords" on page 205
- "Usage notes" on page 206
- "Output fields" on page 206
- "Return, reason, and completion codes" on page 208

- "Examples" on page 209

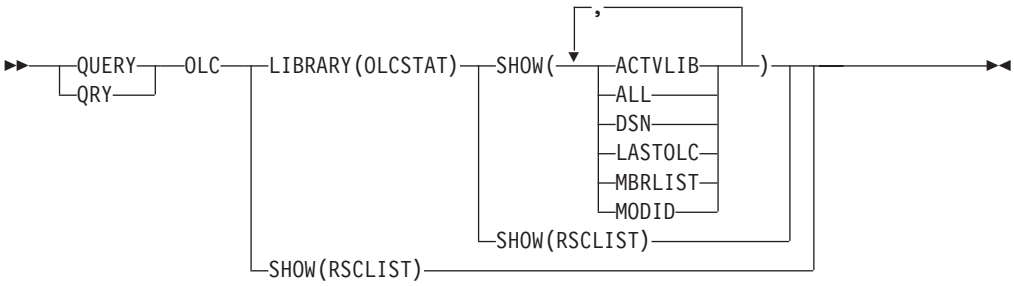
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY OLC command and keywords can be issued.

Table 95. Valid environments for the QUERY OLC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY OLC	X	X	X
LIBRARY	X	X	X
SHOW	X	X	X

Syntax



Keywords

The following keywords are valid for the QUERY OLC command:

LIBRARY
Online change library. The library can be the following:

OLCSTAT
OLCSTAT data set contents.

SHOW()
Specifies the library information returned.

ALL
Returns all the output fields except the RSCLIST information.

ACTVLIB
Displays the suffixed online change library names that are currently active. This includes ACBLIBA or ACBLIBB, FMTLIBA or FMTLIBB, and MODBLKSA or MODBLKSB. These are the online change libraries the IMS online system must use at IMS initialization time.

DSN
OLCSTAT data set name.

LASTOLC
Displays the last online change that was successfully performed. If an IMS was down during the last online change and its restart type does not conflict with the last online change that was performed, it will be permitted to warm start. The last online change type is blank, if no online

changes have been done. The last online change type is one or more of the following, if at least one online change has been done:

- ACBLIB
- FMTLIB
- MODBLKS

MBRLIST
Displays the list of IMS systems that are current with the online change libraries. These are the IMS systems that will be permitted to warm start. The IMS systems either participated in the last online change, or cold started since the last online change.

MODID
Modify ID. The modify ID - 1 represents the number of global online changes that have been performed.

RSCLIST
Returns the ACBLIB members that are copied from the staging ACBLIB to the active ACBLIB. SHOW(RSCLIST) is valid only when TYPE(ACBMBR) OLC is in progress.

Usage notes

The command response is the same for all of the IMS systems. QUERY OLC is valid for an IMS enabled for global online change, but it is not valid for an IMS enabled for local online change. QUERY OLC is not supported on an XRF alternate, an RSR tracker, or an FDBR region. It can only be specified through the OM API.

The QUERY OLC LIBRARY command displays information about global online change, such as the current online change libraries and the IMS systems that are current with the online change libraries. QUERY OLC LIBRARY(OLCSTAT) displays the contents of the global online change status data set, OLCSTAT. Specifying the SHOW keyword may optionally show the current active online change libraries, the list of IMS systems that are current with the online change libraries, the modify ID, and the last online change that was done.

For an IMS running without RM services (RMENV=N), the QUERY OLC command returns information about the local OLCSTAT DS of an IMS system. The command response is different for each IMS because each IMS is required to have a unique OLCSTAT DC. In a no RM environment, if more than one IMS is specified in the route list for the QUERY OLC, only the OLCSTAT DS information for the command IMS master is returned. To obtain OLCSTAT DS information from each IMS that is running without RM, the QUERY OLC command must be issued separately to each IMS. To determine which IMS systems are defined with RMENVNO, issue a QUERY MEMBER SHOW(ATTRIB) command.

The command syntax for this command is defined in XML and is available to automation programs which communicate with OM.

Output fields

The following table shows the QUERY OLC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 96. Output fields for the QUERY OLC command

Short label	Long label	Keyword	Scope	Meaning
ACBL	ACBLIB	ACTVLIB	GBL	The current ACBLIB library. A means that the current ACBLIB library is ACBLIBA. B means that the current ACBLIB library is ACBLIBB.
ADD	ADD	RSCLIST	LCL	Indicates that the resources is being added.
CC	CC	N/A	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified library. Refer to Table 98 on page 209 for more information. The completion code is always returned.
CHG	CHNG	RSCLIST	LCL	Indicates that the resource is being changed.
CPY	COPY	RSCLIST	LCL	Indicates that the resource is being copied.
DBD	DBDName	RSCLIST	LCL	The database or DBD name.
DSN	DSName	DSN	GBL	OLCSTAT data set name.
FMTL	FMTLIB	ACTVLIB	GBL	The current FMTLIB library. A means that the current FMTLIB library is FMTLIBA. B means that the current FMTLIB library is FMTLIBB.
LAST	LastOLC	LASTOLC	GBL	The last successful online change that was successfully performed. If an IMS was down during the last online change and its restart type does not conflict with the last online change that was performed, it will be permitted to warm start. The online change type may include one or more of the following: • ACBLIB • FMTLIB • MODBLKS
LIB	Library	N/A	GBL	Library name. Can be OLCSTAT.

Table 96. Output fields for the QUERY OLC command (continued)

Short label	Long label	Keyword	Scope	Meaning
MBR	MbrName	N/A	N/A	IMSplex member that built output line. IMS identifier of the IMS that build the output. IMS identifier is always returned.
MBRL	MbrList	MBRLIST	GBL	List of IMSplex members that are current with the online change libraries. These are the IMS systems that will be permitted to warm start. The IMS systems either participated in the last online change, or cold started since the last online change.
MODB	MODBLKS	ACTVLIB	GBL	The current MODBLKS and MATRIX libraries. `A` means that the current libraries are MODBLKSA and MATRIXA. `B` means that the current libraries are MODBLKSB and MATRIXB.
MODI	Modid	MODID	GBL	The current modify ID. The modify ID - 1 is the number of successful global online changes that have been performed.
PSB	PSBName	RSCLIST	LCL	The program or PSB name.
RFS	RFSH	RSCLIST	LCL	Indicates the resource is being refreshed.
RSC	RSCName	RSCLIST	LCL	The resource name specified on the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY OLC command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 97. Return and reason codes for the QUERY OLC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY OLC command completed successfully.
X'00000004'	X'00001000'	The QUERY OLC command was not processed on the IMS system as the IMS system is not the command master. No information is returned.
X'00000010'	X'0000400C'	The QUERY OLC command failed because it is invalid for an XRF alternate.
X'00000010'	X'00004014'	The QUERY OLC command failed because it is invalid for an RSR tracker.
X'00000010'	X'0000401C'	The QUERY OLC command failed because it is invalid for an FDBR region.

Table 97. Return and reason codes for the QUERY OLC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'0000410C'	The QUERY OLC command is rejected, because global online change is not enabled. Local online change is enabled. Use the /DISPLAY MODIFY command for local online change. If your IMSplex is made up of some IMS systems that support global online change and some that support local online change, route the QUERY OLC command to an IMS that is enabled for global online change. Issue the QUERY MEMBER TYPE(IMS) SHOW(ATTRIB) command to choose an IMS that has global online change enabled.
X'00000010'	X'00004114'	The QUERY OLC LIBRARY(OLCSTAT) command failed because of an error accessing the OLCSTAT data set. A DFS2843 message is sent to the OM output exit as unsolicited output.
X'00000010'	X'00004118'	The QUERY OLC LIBRARY(OLCSTAT) command failed because of an error allocating the OLCSTAT data set. A DFS2848 message is sent to the OM output exit as unsolicited output.
X'00000010'	X'0000411C'	The QUERY OLC LIBRARY(OLCSTAT) command failed because of an error in the OLCSTAT data set contents. One or more of the values is invalid. A DFS2844 message is sent to the OM output exit as unsolicited output.
X'00000014'	X'00005004'	The QUERY OLC command processing failed because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The QUERY OLC command failed because of an internal IMS error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual library.

The following table contains the completion codes that can be returned on a QUERY OLC command.

Table 98. Completion codes for the QUERY OLC command

Completion code	Meaning
0	The QUERY OLC command completed successfully for the library.

Examples

The following are examples of the QUERY OLC command:

Example 1 for QUERY OLC command

TSO SPOC input:

```
QRY OLC LIBRARY(OLCSTAT) SHOW(ACTVLIB,MODID,MBRLIST)
```

TSO SPOC output:

```
 Response for: QUERY OLC LIBRARY(OLCSTAT) SHOW(ACTVLIB,MODID,MBRLIST)
MbrName    CC Library    ACBLIB   FMTLIB  MODBLKS    Modid MbrList
MS3         0 OLCSTAT         B        A        B        1 IMS3,IMS2,SYS3
```

OM API input:

```
CMD (QRY OLC LIBRARY(OLCSTAT) SHOW(ACTVLIB,MODID,MBRLIST))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 15:22:43.468642</statime>
<stotime>2002.16315:22:45.400709</stotime>
<staseq>B7C4A8029DD62884</staseq>
<stoseq>B7C4A80475885248</stoseq>
<rqsttkn1>USRT011 10082243</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS2    ">
<typ>IMS     </typ>
<styp>DBDC</styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntext>IMS not master, cmd ignored</rsntext>
</mbr>
<mbr name="SYS3    ">
<typ>IMS     </typ>
<styp>DBDC   </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntext>IMS not master, cmd ignored</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMS3    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>OLC  </kwd>
<input>QUERY OLC LIBRARY(OLCSTAT) SHOW(ACTVLIB,MODID,MBRLIST)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"  llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no" len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"   llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT"  align="right" />
<hdr slbl="LIB"  llbl="Library" scope="GBL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="ACBL" llbl="ACBLIB"  scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
<hdr slbl="FMTL" llbl="FMTLIB"  scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
<hdr slbl="MODB" llbl="MODBLKS" scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
<hdr slbl="MODI" llbl="Modid"   scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
<hdr slbl="MBRL" llbl="MbrList" scope="GBL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left"/>
</cmdrsphdr>
<cmdrspdata>
```

```
<rsp>MBR(IMS3    ) CC(   0) LIB(OLCSTAT ) ACBL(B) FMTL(A) MODB(B) MODI(       1)
MBRL(IMS3,IMS2,SYS3) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: QUERY OLC LIBRARY(OLCSTAT) displays the contents of the OLCSTAT data set, which contains global online change status. This command displays the active online change libraries, the modify id, and the list of IMS members that are current with the online change libraries and may therefore warm list. The output shows that the ACBLIBB data set is active, the FMTLIBB data set is active, the MODBLKSB data set is active, and the modify id is 2. SYS3 was the command master that built the output.

Example 2 for QUERY OLC command

TSO SPOC input:

```
QRY OLC LIBRARY(OLCSTAT) SHOW(DSN)
```

TSO SPOC output:

```
MbrName    CC Library  DSName
IMS3        0 OLCSTAT  IMSTESTL.IMS02.OLCSTAT
```

OM API input:

```
CMD (OLC LIBRARY(OLCSTAT) SHOW(DSN))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 15:28:36.353742</statime>
<stotime>2002.16315:28:36.426823</stotime>
<staseq>B7C4A953276CE286</staseq>
<stoseq>B7C4A95339447348</stoseq>
<rqsttkn1>USRT011 10082836</rqsttkn1>
<rc>0200000C</rc><rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS2    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntext>IMS not master; cmd ignored</rsntext>
</mbr>
<mbr name="SYS3    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntext>IMS not master; cmd ignored</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMS3    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>OLC  </kwd>
<input>QUERY OLC LIBRARY(OLCSTAT) SHOW(DSN)</input>
</cmd>
<cmdrsphdr>
```

```
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"  llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT"  align="right" />
<hdr slbl="LIB" llbl="Library" scope="GBL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="DSN" llbl="DSName"  scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS3   ) CC(   0) LIB(OLCSTAT ) DSN(IMSTESTL.IMS02.OLCSTAT) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: This QUERY OLC command displays the OLCSTAT data set name. IMS3 was the command master that built the output.

Example 3 for QUERY OLC command

TSO SPOC input:

```
QUERY OLC SHOW(RSCLIST)
```

TSO SPOC output:

```
MbrName    CC DBDName  PSBName  ADD
IMS1        0 OLCDB105          Y
IMS2        0 OLCDB105          Y
IMS1        0 OLCDB111          Y
IMS2        0 OLCDB111          Y
IMS1        0 OLCDI111          Y
IMS2        0 OLCDI111          Y
IMS1        0 OLCDX111          Y
IMS2        0 OLCDX111          Y
IMS1        0          OLCPB105 Y
IMS2        0          OLCPB105 Y
IMS1        0          OLCPB111 Y
IMS2        0          OLCPB111 Y
```

OM API input:

```
CMD (QUERY OLC SHOW(RSCLIST))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.268 18:28:28.326510</statime>
<stotime>2006.268 18:28:28.341969</stotime>
<staseq>BF75BA3F0266E263</staseq>
<stoseq>BF75BA3F062D136E</stoseq>
<rqsttkn1>USRT001 10112828</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>OLC             </kwd>
<input>QUERY OLC SHOW(RSCLIST) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
```

```
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RSC" llbl="RSCName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DBD" llbl="DBDName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PSB" llbl="PSBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ADD" llbl="ADD" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CHG" llbl="CHNG" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CPY" llbl="COPY" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RFS" llbl="RFSH" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS2    ) CC(   0) DBD(OLCDB105) ADD(Y) </rsp>
<rsp>MBR(IMS2    ) CC(   0) DBD(OLCDX111) ADD(Y) </rsp>
<rsp>MBR(IMS2    ) CC(   0) DBD(OLCDB111) ADD(Y) </rsp>
<rsp>MBR(IMS2    ) CC(   0) DBD(OLCDI111) ADD(Y) </rsp>
<rsp>MBR(IMS2    ) CC(   0) PSB(OLCPB105) ADD(Y) </rsp>
<rsp>MBR(IMS2    ) CC(   0) PSB(OLCPB111) ADD(Y) </rsp>
<rsp>MBR(IMS1    ) CC(   0) DBD(OLCDB105) ADD(Y) </rsp>
<rsp>MBR(IMS1    ) CC(   0) DBD(OLCDX111) ADD(Y) </rsp>
<rsp>MBR(IMS1    ) CC(   0) DBD(OLCDB111) ADD(Y) </rsp>
<rsp>MBR(IMS1    ) CC(   0) DBD(OLCDI111) ADD(Y) </rsp>
<rsp>MBR(IMS1    ) CC(   0) PSB(OLCPB105) ADD(Y) </rsp>
<rsp>MBR(IMS1    ) CC(   0) PSB(OLCPB111) ADD(Y) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: This example shows the output from a QRY OLC SHOW(RSCLIST) command after an INIT OLC PHASE(PREPARE) TYPE(ACBMBR) command has been issued.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY OLREORG command

Use the QUERY OLREORG command to query information about an online reorganization of a HALDB partition.

Information that is returned by the QUERY OLREORG command includes the status, rate, and number of bytes that have been moved.

Reorganizing a database means modifying the structure of a database or changing how the data in the database is organized in order to improve performance.

Subsections:

- "Environment" on page 214
- "Syntax" on page 214

- "Keywords"
- "Usage notes" on page 216
- "Output fields" on page 216
- "Return, reason, and completion codes" on page 217
- "Example" on page 218

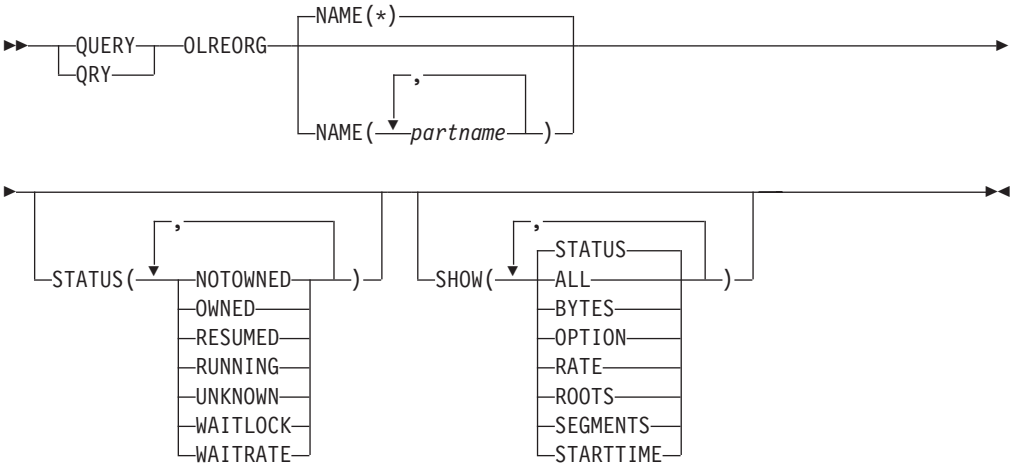
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY OLREORG command and keywords can be issued.

Table 99. Valid environments for the QUERY OLREORG command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY OLREORG	X	X	
NAME	X	X	
STATUS	X	X	
SHOW	X	X	

Syntax



Keywords

The following keywords are valid for the QUERY OLREORG command:

NAME()
Specifies the names of the HALDB PHDAM or PHIDAM partition to be queried. NAME() is optional. A parameter with the wildcard character (*) is not allowed, except as NAME(*) for all defined HALDB partitions. NAME(*) is the default.

SHOW()
Specifies the output fields to return. The default is SHOW(STATUS).

ALL
Returns all of the following output fields on each response line.

BYTES
Returns the total number of bytes that have been moved to the output data set.

OPTION
Returns the current option, which can be one of the following:

- DEL - The output data sets will be deleted at the end of OLR
- NODEL - The output data sets will not be deleted at the end of OLR

RATE
Returns the rate at which the HALDB OLR is running, from 1 to 100.

ROOTS
Returns the total number of roots that have been moved to the output data set.

SEGMENTS
Returns the total number of segments that have been moved to the output data set.

STARTTIME
Returns the local time when OLR was started.

STATUS
Returns online reorganization status.

STATUS()
Displays online reorganizations that possess at least one of the specified statuses. If the STATUS keyword is not specified, any online reorganization with a status of RUNNING, OWNED, NOTOWNED, WAITRATE, or WAITLOCK is returned.

NOTOWNED
Specifies that the output is for the HALDB OLRs that have been temporarily stopped by the TERMINATE OLREORG command and, therefore, are not owned by any IMS.

To inquire on OLRs that have been terminated using the TERMINATE OLREORG command, use the commands QUERY OLREORG STATUS(NOTOWNED) with ROUTE(*) on the command request and /RMLIST DBRC='DB DBD(partname)'. If all systems on the PLEX show status, then OLR has been terminated.

OWNED
Specifies that the output is for HALDB OLRs that are owned by any IMS. The OLRs that are running on the IMS where the command is being processed displays a STATUS of RUNNING. OLRs running on other IMS subsystems displays a STATUS of OWNED.

RESUMED
OLR is resumed after being stopped for some reason such as a TERM OLREORG command or a user abend.

RUNNING
Specifies the output is for the HALDB OLRs that are owned by each IMS for the specified partname or partnames. You can use this keyword to determine which IMS has an online reorganization running for a given partname.

UNKNOWN
Specifies that the output is for those part names on each IMS for which the status of the HALDB OLR cannot be determined. This inability to

determine the status can be caused by situations such as the HALDB master or partition being taken offline by a /DBR DB command, the IMS not being authorized to the named partname because of an IRLM failure, or the partition has not been accessed.

WAITLOCK
OLR is waiting for a lock.

WAITRATE
OLR is waiting because of an intentional delay. This intentional delay was caused because a value of less than 100 was specified on the RATE parameter.

Usage notes

You can issue the QUERY OLREORG command only from the OM API. Responses from each IMS to which the command was routed are consolidated by OM. If you specify names on the NAME parameter, response lines are returned for each specified part name. For the default parameter, NAME(*), response lines are returned only for the HALDB partitions that have online reorganizations in progress at each IMS.

A nonzero return code and a nonzero reason code are returned when the command is routed to an XRF alternate system and when the command is routed to an RSR tracking system.

The output for this command is defined in XML and is available to automation programs that communicate with OM.

The QRY OLREORG command can be issued to obtain OLR statistics for an OLR that has been stopped for some reason such as a TERM OLREORG command or a user abend. The data associated with the terminated OLR will be maintained and provided under any of the following conditions:

- The TERMINATE OLR command is issued.
- Abnormal OLR termination occurs (for example, DFS2971W message is issued)
- IMS is normally shut down

Note: If the partition or HALDB master has been taken offline with a /DBR command, the status will not be available through the QUERY OLREORG command until the OLR is resumed. The status, however, can be obtained through the LIST.DB or LIST.RECON command.

Output fields

The following table shows the QUERY OLREORG output fields. The columns in the table are as follows:

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the long label generated in the XML output.

Show keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 100. Output fields of QUERY OLREORG

Short label	Long label	Show keyword	Meaning
BYTES	Bytes-Moved	BYTES, ALL	Total number of bytes moved.
CC	CC	N/A	Completion code.
LSTT	LclStat	N/A	Status of HALDB OLR.
MBR	MbrName	N/A	The IMS from which the command was issued.
OPT	Option	OPTION, ALL	Option specified on the INITIATE OLREORG or UPDATE OLREORG command.
PART	Partition	N/A	Partition name.
RATE	Rate	RATE, ALL	The speed at which HALDB OLR runs. A value of 1 to 100 percent.
RESM	Resumed	N/A	RESUMED status if HALDB OLR is resumed.
ROOTS	Roots-Moved	ROOTS, ALL	Total number of roots moved.
SEGS	Segs-Moved	SEGMENTS, ALL	Total number of segments moved.
STRTT	StartTime	STARTTIME, ALL	The time OLR started as recorded in the RECON data set.

Return, reason, and completion codes

The OM return and reason codes that might be returned as a result of the QUERY OLREORG command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 101. Return and reason codes for the QUERY OLREORG command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY OLREORG command completed successfully.
X'00000004'	X'00001010'	No matches found for filter.
X'00000008'	X'00002004'	Invalid command keyword.
X'0000000C'	X'00003000'	At least one request was successful.
X'0000000C'	X'00003004'	None of the requests was successful.
X'00000010'	X'00004030'	Command is invalid for LSO=Y.
X'00000010'	X'00004014'	Command was issued on an RSR tracker.
X'00000010'	X'0000400C'	Command was issued on an XRF alternate.
X'00000010'	X'00004200'	The QUERY OLREORG command is not processed because IMS shutdown is in progress.
X'00000014'	X'00005000'	A GETMAIN error occurred.

The following table includes an explanation of the completion codes. Errors unique to the processing of QUERY OLREORG command are returned as completion codes. A completion code is returned for each action against a HALDB partition.

Table 102. Completion codes for the QUERY OLREORG command

Completion code	Meaning
0	The QUERY OLREORG command completed successfully for the partition.
10	Resource name is invalid.
14	Resource is not a partition name.
1C	Resource is a partitioned secondary index.
24	No HALDB OLR is in progress.
28	No DMB is loaded.
CB	Partition is not in specified status.

Example

In this example, the QUERY OLREORG command is routed to IMSA. The command is issued to obtain the information about all of the OLRs that are in progress at IMSA. The output that is returned contains the following information:

- The partition name
- The IMSID
- The status of OLRs in progress
- The rate of OLR
- The number of bytes moved

TSO SPOC input:

```
QRY OLREORG NAME(*) SHOW(ALL)
```

TSO SPOC output:

```
Partition MbrName    CC LclStat  Rate  Bytes-Moved Segs-Moved...
POHIDKA   IMS1        0 RUNNING   100        15678         97...
PVHDJ5A   IMS1        0 RUNNING   100         4630         29...

... Roots-Moved  Option   Resumed StartTime
...          11  NODEL    Y       2007.296 10:20:21.61
...           5  DEL              2007.296 10:20:21.84
```

OM API input:

```
CMD (QRY OLREORG NAME(*) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2007.296 17:43:42.714976</statime>
<stotime>2007.296 17:43:42.715488</stotime>
<staseq>C163CD37F5860D82</staseq>
<stoseq>C163CD37F5A60342</stoseq>
<rqsttkn1>USRT011 10104342</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
```

```
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>OLREORG         </kwd>
<input>QRY OLREORG NAME(*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PART" llbl="Partition" scope="lcl" sort="a" key="1"
 scroll="no" len="7" dtype="char" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="lcl" sort="n" key="0"
scroll="no"
 len="8" dtype="char" align="left" />
<hdr slbl="CC" llbl="CC" scope="lcl" sort="n" key="0"
scroll="yes"
 len="4" dtype="int" align="right" />
<hdr slbl="LSTT" llbl="LclStat" scope="lcl" sort="n" key="0"
 scroll="yes" len="*" dtype="char" align="left" />
<hdr slbl="RATE" llbl="Rate" scope="lcl" sort="n" key="0"
scroll="yes"
 len="3" dtype="int" align="right" skipb="yes" />
<hdr slbl="BYTES" llbl="Bytes-Moved" scope="lcl" sort="n"
key="0"
 scroll="yes" len="12" dtype="int" align="right" skipb="yes" />
<hdr slbl="SEGS" llbl="Segs-Moved" scope="lcl" sort="nO" key="0"
 scroll="yes" len="10" dtype="int" align="right" skipb="yes" />
<hdr slbl="ROOTS" llbl="Roots-Moved" scope="lcl" sort="n"
key="0"
 scroll="yes" len="10" dtype="int" align="right" skipb="yes" />
<hdr slbl="OPT" llbl="Option" scope="lcl" sort="n" key="0"
scroll="yes"
 len="8" dtype="char" align="left" skipb="yes" />
<hdr slbl="RESM" llbl="Resumed" scope="lcl" sort="n" key="0"
 scroll="yes" len="1" dtype="char" align="left" skipb="yes" />
<hdr slbl="STRTT" llbl="StartTime" scope="lcl" sort="n" key="0"
 scroll="yes" len="20" dtype="char" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp> PART(POHIDKA ) MBR(IMS1    ) CC(   0) LSTT(RUNNING )
RATE(100)
 BYTES(       15678) SEGS(        97) ROOTS(        11)
 OPT(DEL     )
 RESM(Y) STRTT(2007.296 10:20:21.61) </rsp>
<rsp> PART(PVHDJ5A ) MBR(IMS1    ) CC(   0) LSTT(RUNNING )
RATE(100)
 BYTES(        4630) SEGS(        29) ROOTS(         5)
 OPT(DEL     )
 STRTT(2007.296 10:20:21.84) </rsp>
</cmdrspdata>
</imsout>
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY OTMADESC command

Use the QUERY OTMADESC command to query information about an existing IMS Open Transaction Manager Access (OTMA) destination descriptor.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 221
- "Usage notes" on page 222
- "Output fields" on page 223
- "Return, reason, and completion codes" on page 224
- Examples

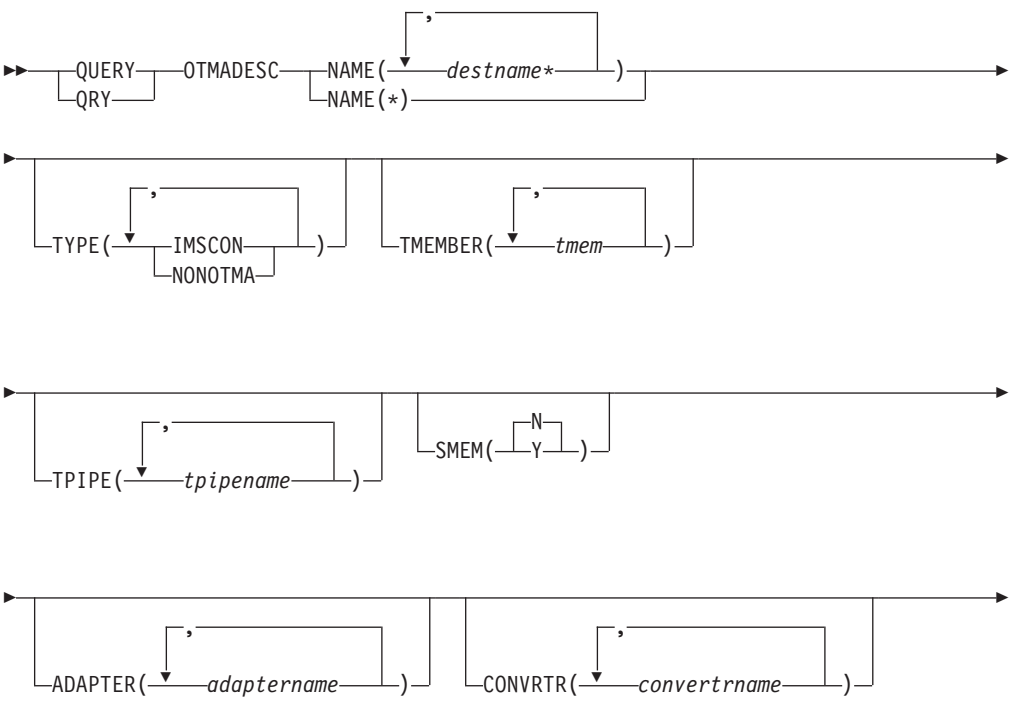
Environment

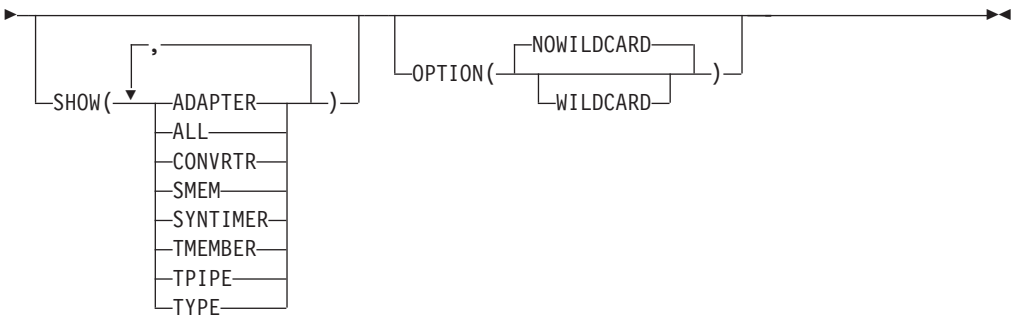
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and its keywords.

Table 103. Valid environments for the QUERY OTMADESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY OTMADESC	X		X

Syntax





Keywords

The following keywords are valid for the QUERY OTMADESC command:

ADAPTER(*adaptername***)**
A 1- to 8-character name that identifies the IMS Connect adapter. This keyword is an optional parameter for TYPE(IMSCON). When an adapter name is specified, information is displayed from only the descriptors that specify that adapter name. This keyword is ignored if TYPE(NONOTMA) is specified.

CONVRTR(*convertrname***)**
A 1- to 8-character converter name associated with the adapter specified on the ADAPTER keyword. When a converter name is specified, information is displayed from only the descriptors that specify that converter name.

NAME(*destname**)**
A 1- to 8-character destination name that identifies the OTMA destination descriptors to display. This keyword is required. If OPTION(WILDCARD) is also specified, an asterisk can be used as a wildcard character.

OPTION(NOWILDCARD | WILDCARD)
Determines whether an asterisk in the NAME keyword is treated as a wildcard character. When NOWILDCARD is specified, an asterisk is processed as a normal character and a descriptor is displayed only if the destination name coded in the descriptor in the DFSYDTx PROCLIB member includes a matching asterisk character. When WILDCARD is specified, an asterisk in the NAME keyword is treated as wildcard character and any descriptors that have a destination name that matches the characters not masked by the asterisk are displayed.

SHOW()
Specifies which information to display in the output fields.

ADAPTER
Displays the name that identifies the IMS Connect adapter.

ALL
Displays all available information.

CONVRTR
Displays the converter name used by the adapter.

SMEM
Displays the super member indicator.

SYNTIMER
Displays the timeout value for synchronous callout processing.

TMEMBER
Displays the name of TMEMBER or the Super Member if SMEM indicates as such.

TPIPE
Displays the TPIPE name under a TMEMBER.

TYPE
Displays the type of descriptor.

SMEM(Y | N)
An optional parameter that specifies whether the TMEMBER name is a super member. When SMEM is specified, information is displayed from only the descriptors that specify the matching Y or N value for SMEM. SMEM and TYPE(NONOTMA) are mutually exclusive.

TMEMBER(*tmem*)
An optional parameter that is used to filter by a 1- to 16-character TMEMBER name. When a tmember name is specified, information is displayed from only the descriptors that specify that tmember name. TMEMBER and TYPE(NONOTMA) are mutually exclusive.

TPIPE(*tpipename*)
An optional parameter that is used to filter by a 1- to 8-character TPIPE name. When a tpipe name is specified, information is displayed from only the descriptors that specify that tpipe name. TPIPE and TYPE(NONOTMA) are mutually exclusive.

TYPE(IMSCON | NONOTMA)
An optional keyword that can be either IMSCON or NONOTMA. Both output types are displayed if the keyword is not specified. To filter the display, specify either IMSCON or NONOTMA. To include the other keywords, such as TMEMBER, TPIPE, SMEM, ADAPTER, or CONVRTR, use the SHOW keyword. If TYPE(NONOTMA) is specified, however, the rest of the display parameters will show blanks even if the SHOW keyword is specified.

Usage notes

The QUERY OTMADESC command is used to display the values specified in the destination routing descriptors that are identified in the NAME() keyword. The results returned by the QUERY OTMADESC command can be filtered by secondary keywords, such as TPIPE(). For example, when QUERY OTMADESC NAME(*abc**) TPIPE(*xyz*) OPTION(WILDCARD) is specified, information is returned for only descriptors that have a destination name that starts with *abc* and a tpipe name of *xyz*.

Use the SHOW() keyword to control what information is displayed from the OTMA destination descriptors found by the command. For example, if you include SHOW(TMEMBER) in the command, the TMEMBER values are returned.

The QUERY OTMADESC command does not process asterisks as wildcard characters. However, you can use an asterisk as a wildcard character in the NAME() keyword if you specify OPTION(WILDCARD) when you issue the command.

No log records are written when the command is issued.

The QUERY OTMADESC command can be issued on an Extended Recovery Facility (XRF) alternate or a remote site recovery (RSR) tracking environment. The

information retrieved might differ from the active system because of timing issues. For example, take the scenario where the UPDATE OTMADESC command is issued on the active system, the log record is written, and the QUERY OTMADESC command is issued on both the active and alternate system. The information about the active system reflects the current information whereas the alternate system reflects the old information. This is caused by the log record being read and updated in the alternate system.

You can issue a valid QUERY OTMADESC command that does not return any results because one of the filter keywords excluded the entire result set. For example, querying descriptors with both the ADAPTER keyword and the MQFORMAT keyword will never return any results because those keywords are never used together in a valid descriptor. If your query does not return any results, ensure that no mutually exclusive filtering keywords were specified.

Output fields

The following table shows the QUERY OTMADESC output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 104. Output fields for the QUERY OTMADESC command

Short label	Long label	Keyword	Meaning
ADAP	Adapter	ADAPTER	Adapter name.
CC	CC	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned.
CCTXT	CCText	N/A	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
CVRTR	Converter	CONVRTR	Converter name.
DEST	DestName	NAME	Destination name.
MBR	MbrName	N/A	Member name.
SMEM	SMem	SMEM	Indicator that shows whether the destination is a super member.
SYNTO	Syntimer	SYNTIMER	Timeout value for synchronous callout processing.
TMEM	TMember	TMEMBER	OTMA TMEMBER name.
TPIPE	TPipe	TPIPE	TPIPE name.
TYPE	Type	TYPE	Output type.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY OTMADESC command. The OM return and reason codes that might be returned as a result of the QUERY OTMADESC command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 105. Return and reason codes for the QUERY OTMADESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See Table 106 for details.
X'02000008'	X'00002000'	The command contains an invalid verb or no client is registered for the verb.
X'02000008'	X'00002004'	The command contains an invalid primary keyword or no client is registered for the keyword.
X'02000008'	X'00002028'	The command contains an invalid keyword.
X'02000008'	X'0000202C'	The command contains an unknown positional parameter.
X'02000008'	X'00002034'	The command contains an incomplete keyword parameter.
X'02000008'	X'00002038'	The command is missing a required parameter.
X'02000008'	X'0000203C'	The command contains an invalid keyword parameter value.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 106. Completion codes for the QUERY OTMADESC command

Completion code	Completion code text	Meaning
0	Command completed successfully	The QUERY OTMADESC command completed successfully for the resource.
165	No Desc found meet any criteria	No descriptors were found based on the filters specified.
166	No entries found	Command failed for QUERY OTMADESC, because no entries were found in the table of destination routing descriptors.

Example 1 for QUERY OTMADESC command

The following are examples of the QUERY OTMADESC command:

Example 1 for QUERY OTMADESC command

TSO SPOC input:

```
QUERY OTMADESC NAME(OTMACL*) SHOW(TYPE,TMEMBER,SMEM)
```

TSO SPOC output:

```
DestName MbrName  CC Type    TMember  SMem
OTMACL*  IMSA      0 IMSCON  HWS2     N
```

Explanation: An asterisk is specified in the name specified in the NAME() keyword, but because the default value for the OPTION keyword is NOWILDCARD and OPTION(WILDCARD) is not specified, the QUERY command displays only OTMACL* and not OTMACL99. It treats OTMACL* as a stand-alone entry in the table of destination routing descriptors.

Example 2 for QUERY OTMADESC command

TSO SPOC input:

```
QUERY OTMADESC NAME(OTMACL*) SHOW(ALL) OPTION(WILDCARD)
```

TSO SPOC output:

```
DestName MbrName  CC Type    TMember  TPipe     SMem
OTMACL99 IMSA      0 IMSCON  HWS1     HWS1TP01  N
OTMACL*  IMSA      0 IMSCON  HWS2               N
```

Explanation: The QUERY command will display both OTMACL99 and OTMACL*, because the OPTION(WILDCARD) keyword is specified. It treats OTMACL* as a mask for a group of names including OTMACL99 and, the stand-alone entry OTMACL* in the table of destination routing descriptors.

Example 3 for QUERY OTMADESC command

TSO SPOC input:

```
QUERY OTMADESC NAME(OTMD*) TYPE(IMSCON) SHOW(TMEMBER,SMEM,SYNTIMER) OPTION(WILDCARD)
```

TSO SPOC output:

```
DestName MbrName    CC Type     TMember          SMem Syntimer
OTMD*    IMS1        0 IMSCON   HWS1             N      1000
OTMDSC01 IMS1        0 IMSCON   HWS1             N      3500
OTMDSC02 IMS1        0 IMSCON   HWS1             N    999999
OTMDSC03 IMS1        0 IMSCON   HWS1             N      1000
OTMDSC04 IMS1        0 IMSCON   SM01             Y      4000
OTMDSC08 IMS1        0 IMSCON   HWS1             N
```

Explanation: The last descriptor, OTMDSC08, does not show any `Syntimer` value because this descriptor is normally not used for synchronous callout. `Syntimer` is only for synchronous callout processing.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY OTMATI command

Use the QUERY OTMATI command to display information about IMS OTMA message workload.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 227
- "Usage notes" on page 229
- "Output fields" on page 230
- "Return, reason, and completion codes" on page 231
- "Examples" on page 232

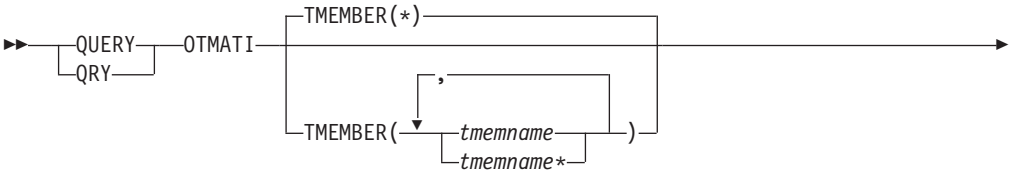
Environment

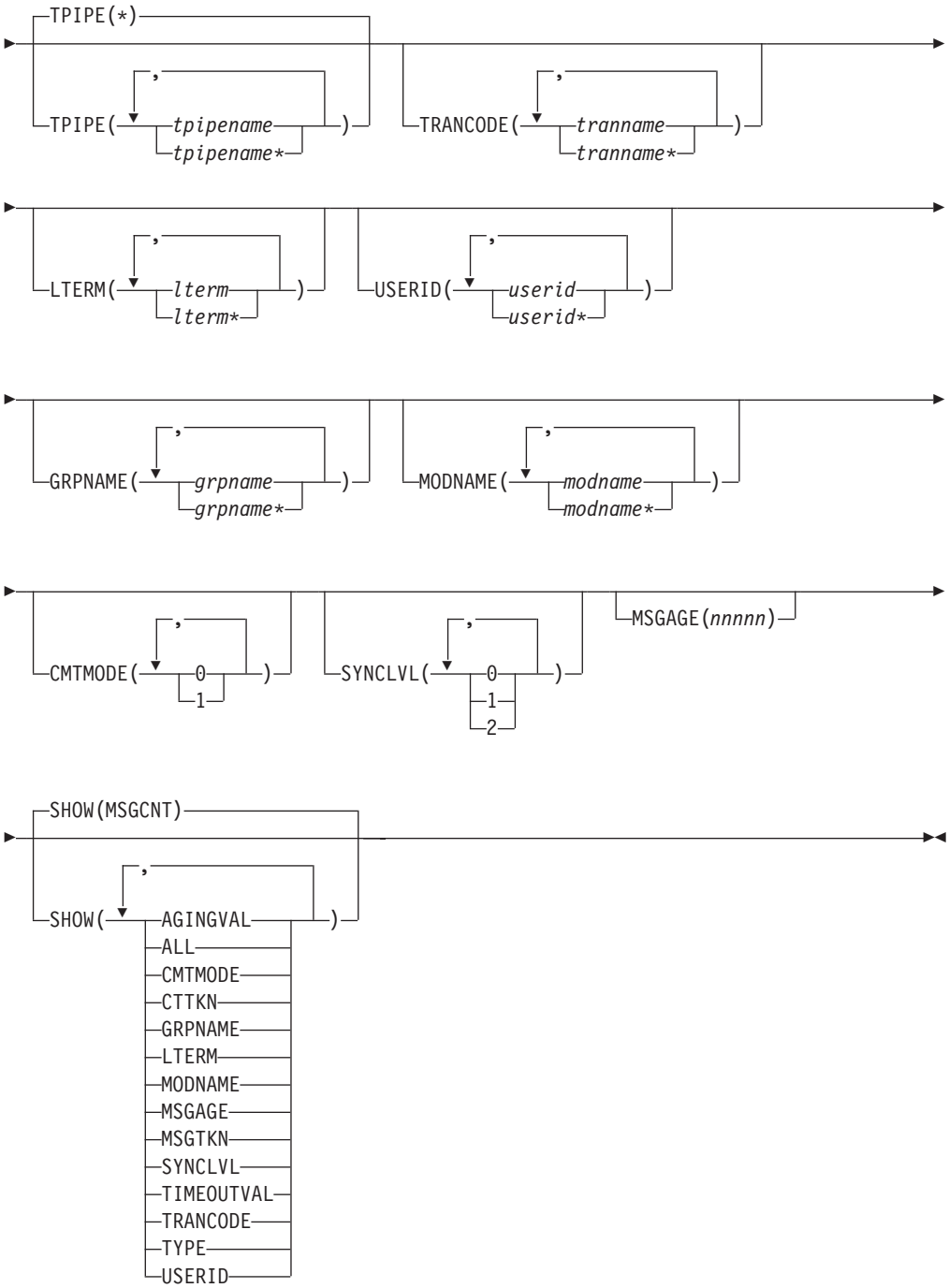
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 107. Valid environments for the QUERY OTMATI command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY OTMATI	X		X
CMTMODE	X		X
GRPNAME	X		X
LTERM	X		X
MODNAME	X		X
MSGAGE	X		X
SHOW	X		X
SYNCLVL	X		X
TMEMBER	X		X
TPIPE	X		X
TRANCODE	X		X
USERID	X		X

Syntax





Keywords

The following keywords are valid for the QUERY OTMATI command:

CMTMODE
Specifies the commit mode of the workload to be displayed. Specify 0 to display all workloads that are in commit-then-send (CM0) mode, which is supported on both persistent and transaction sockets and supports only synch level CONFIRM. Specify 1 to display all workloads that are in send-then-commit (CM1) mode, which is supported on both persistent and transaction sockets and supports synch level NONE, CONFIRM, and SYNCH.

GRPNAME
Specifies a 1- to 8-character RACF Group ID that is included in the security prefix of the message. Any group name that exceeds 8 characters is flagged as an error.

LTERM
Specifies a 1- to 8-character override LTERM name that is included in the OTMA message state prefix. Any LTERM name that exceeds 8 characters is flagged as an error.

MODNAME
Specifies a 1- to 8-character message output descriptor name that is included in the OTMA message state prefix. This descriptor is associated with the transaction or the program to be scheduled.

MSGAGE
Specifies the minimum amount of clock time since the message (YTIB) became active. The MSGAGE value has a range of 1 through 86400. This value represents the number of seconds within a 24-hour period. Any values outside of this range is an error.

TMEMBER
Specifies a 1- to 16-character OTMA target member name. The member is a client of OTMA, such as IMS Connect.

TPIPE
Specifies a 1- to 8-character OTMA transaction pipe name.

TRANCODE
Specifies a 1- to 8-character transaction code associated with the program that is scheduled.

SYNCLVL
Specifies the synch level. Specify 0 for a synch level of NONE, which requires no acknowledgment from the client. Specify 1 for a synch level of CONFIRM, which requires the client to acknowledge delivery of output messages. Specify 2 for a synch level of SYNCH, for two-phase commit processing that involves multiple participants in sync point processing managed through z/OS Resource Recovery Services (RRS).

USERID
Specifies a 1- to 8-character RACF user ID that is included in the security prefix of the message.

SHOW
Specifies the output fields to be returned. The filters supported with the SHOW keyword are:

AGINGVAL
Displays the aging value (how often the cached user ID accessor environment element (ACEE) should be refreshed). The aging value is either the message aging value or the client aging value and comes from the message control prefix or from the message state prefix, respectively.

ALL
Displays all information about the OTMA message workload.

CMTMODE
Displays the commit mode. 0 represents “Commint-then-send” mode, and 1 represents “Send-then-commit” mode.

CTTKN
Displays the context token when a transaction is in a two-phase commit that involves multiple participants in sync point processing managed through RRS.

GRPNAME
Displays the RACF group ID.

LTERM
Displays the override LTERM name.

MODNAME
Displays the override MODNAME.

MSGAGE
Displays the minimum age in seconds since the message (TIB) became active.

MSGCNT
Displays the total number of active TIBs associated with messages, depending on the various parameter values requested.

MSGTKN
Displays the correlator token.

TIMEOUTVAL
Displays the timeout value for CM1 that is missing ACK.

TRANCODE
Displays the transaction code that is associated with the message.

TYPE
Displays the message type, such as SMB transaction, CPIC transaction, IMS command, message switch, message recoverable, conversational transaction, or response.

SMB SMB transaction
CPC CPIC transaction
CMD IMS command
APC Message switch
RCV Recoverable transaction
CON Conversational transaction
EMH Fast Path transaction
RSP Transaction response

USERID
Displays the user ID that is included in the security data prefix of the message.

SYNCLVL
Displays the synch level of NONE, CONFIRM, or SYNCH.

Usage notes

The QUERY OTMATI command can be issued both in the active and the alternate system, whether it be on an XRF alternate or RSR tracker system.

The QUERY OTMATI command can be specified through the OM API, including the TSO SPOC and IBM Management Console for IMS and DB2® for z/OS (Management Console) user interface. The output of this command is displayed also in the TSO SPOC or in Management Console.

If the QUERY OTMATI command is issued without the SHOW keyword, the display will show the workloads on each of the IMS instances in the sysplex together with the TMEMBER name, TPIPE name, and the total number of active messages. If any keywords other than SHOW are specified, those keywords will be used as a filter to display the total number of messages in the queue as a subset of the total workload. If the SHOW keyword is specified, the total number of active messages will not be displayed; instead, each individual active message will be displayed together, filtered by the other specified keywords. No log records are written.

Output fields

The following table shows the QUERY OTMATI output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 108. Output fields for the QUERY OTMATI command

Short label	Long label	Keyword	Meaning
AGINGVAL	AgingVal	AGINGVAL	RACF ACEE aging value. The value can be: • a number between 0 and 99999 (seconds) • ">99999" (the default) • a blank, which indicates that the TIB was processed in the back-end system
CC	CC	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 231 for more information. The completion code is always returned.
CCTXT	CCText	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.

Table 108. Output fields for the QUERY OTMATI command (continued)

Short label	Long label	Keyword	Meaning
CMTMODE	CmtMode	CMTMODE	Commit mode. The value can be one of the following: **0** Commit mode 0 **1** Commit mode 1 **1*** Transaction instance block (TIB) is processed in the back-end system and is always a commit mode 1 transaction.
CTTKN	CtTkn	CTTKN	Context token.
GRPNAME	GrpName	GRPNAME	RACF group name.
LTERM	Lterm	LTERM	Override LTERM name.
MBR	MbrName	N/A	IMSplex member that built the output line. The IMS identifier is always returned.
MODNAME	MODname	MODNAME	Override MODNAME.
MSGAGE	MsgAge	MSGAGE	Age of the message in the system. The value can be a number between 0 and 86400. For TIBs that are in the system for more than 86400 seconds, or one day, ">86400" is displayed.
MSGCNT	MsgCnt	MSGCNT	Number of entries that fall in the same filtering criteria. The value can be a number between 0 and 99999, or ">99999", which indicates that the number of TIBs exceeds five digits.
MSGTKN	MsgTkn	MSGTKN	Client token.
MSGTYP	MsgType	TYPE	Message type.
SYNCLVL	SyncLvl	SYNCLVL	Synch level.
TMEM	TMember	TMEM	TMEMBER name.
TMOVAL	TimeoutVal	TIMEOUTVAL	Timeout value for CM1 that is missing ACK. The value can be a number between 0 and 255, or a blank, which indicates that the TIB was processed in the back-end system.
TPIPE	TPipe	TPIPE	Tpipe name.
TRAN	Trancode	TRAN	Transaction name.
UID	Userid	USERID	User ID.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY OTMATI command. The OM return and reason codes that may be returned as a result of the QUERY OTMATI command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 109. Return and reason codes for the QUERY OTMATI command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See Table 110 for details.
X'02000008'	X'00002000'	The command contains an invalid verb or no client is registered for the verb.
X'02000008'	X'00002004'	The command contains an invalid primary keyword or no client registered for the keyword.
X'02000008'	X'00002028'	The command contains an invalid keyword.
X'02000008'	X'0000202C'	The command contains an unknown positional parameter.
X'02000008'	X'00002034'	The command contains an incomplete keyword parameter.
X'02000008'	X'00002038'	The command is missing a required parameter.
X'02000008'	X'0000203C'	The command contains an invalid keyword parameter value.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 110. Completion codes for the QUERY OTMATI command

Completion code	Completion code text	Meaning
0	Command completed successfully	The QUERY OTMATI command completed successfully for the resource.
4	Command completed unsuccessfully	The QUERY OTMATI command failed.

Examples

The following are examples of the QUERY OTMATI command:

Example: QUERY OTMATI

TSO SPOC input:

```
QUERY OTMATI
```

TSO SPOC output:

```
TMember     TPipe        MbrName   CC    MsgCnt
MQ          APPLA        IMSA       0      102
MQ          APPLB        IMSB       0      201
IMSB        WAS          APPLC      0      301
```

Explanation: With the QUERY OTMATI command with no parameters, the display shows the active OTMA send-then-commit messages represented by transaction instance blocks (YTIBs) that are currently running , waiting for a response, or both. TI stands for transaction instance for OTMA.

Example: QUERY OTMATI MSGAGE(8)

TSO SPOC input:

```
QRY OTMATI MSGAGE(8)
```

TSO SPOC output:

```
TMember    TPipe       MbrName    CC    MsgCnt    MsgAge
HWS1       TPIPE1      IMSA        0         2         8
HWS2       TPIPE2      IMSA        0         3         9
```

Explanation: If the QUERY OTMATI command is issued with the MSGAGE keyword, the display will show a subset of the display of Example #1 that has messages with an age of 8 seconds or more. The MsgAge column shows the minimum age found for messages in that subset. For example, the output shows that a subset of the two messages for TPIPE(TPIPE1) contains messages that have an age of 8 seconds or more. The three messages for TPIPE (TPIPE2) contain messages that also have an age of 8 seconds or more and the least age found for its messages was 9 seconds. The display can further be filtered using any keywords, such as TRANCODE and USERID.

Example: QUERY OTMATI MSGAGE(8) SHOW(MODNAME)

TSO SPOC input:

```
QUERY OTMATI MSGAGE(8) SHOW(MODNAME)
```

TSO SPOC output:

```
TMember     TPipe      MbrName     CC    MsgAge     Userid
HWS1        TPIPE1     IMSA         0        10     SVL01
HWS1        TPIPE1     IMSA         0         8     SVL08
HWS2        TPIPE2     IMSA         0         9     IMS02
HWS2        TPIPE2     IMSA         0        16     IMS07
HWS2        TPIPE2     IMSA         0        11     IMS08
```

Explanation: If the QUERY OTMATI command is issued with a MSGAGE keyword and a SHOW keyword specifying MODNAME, the display will expand the display of Example #2 but without the MsgCnt column. Instead of grouping the messages as in Example #2, each message is displayed separately.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY PGM command

Use the QUERY PGM command to query information about program resources.

A program resource defines the application program requirements for application programs that run under the control of the DB/TM environment, as well as for application programs that access databases through DBCTL.

Subsections:

- "Environment" on page 234
- "Syntax" on page 234
- "Keywords" on page 235
- "Usage notes" on page 237

- "Equivalent IMS type-1 commands" on page 238
- "Output fields" on page 238
- "Return, reason, and completion codes" on page 244
- "Examples" on page 245

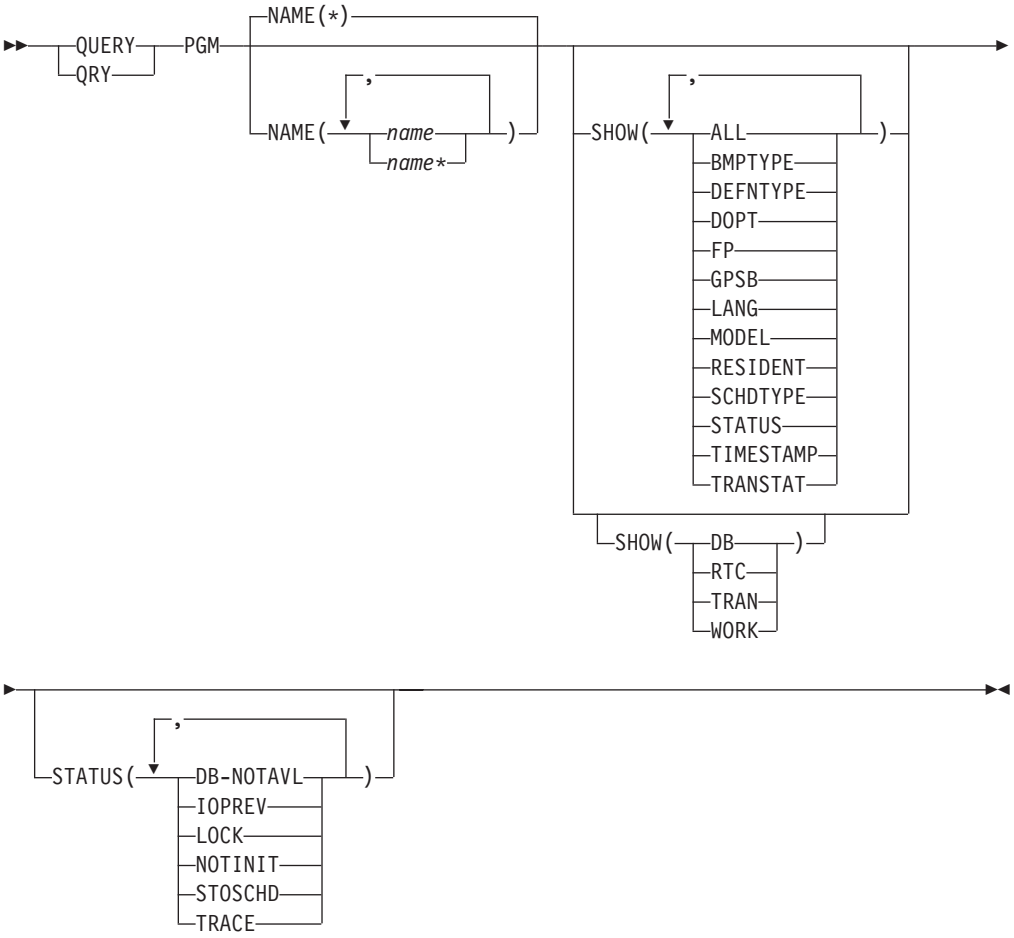
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 111. Valid environments for the QUERY PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY PGM	X	X	X
NAME	X	X	X
SHOW	X	X	X
STATUS	X	X	X

Syntax



Keywords

The following keywords are valid for the QUERY PGM command:

NAME
Specifies the 1-8 character name of the program. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*) which returns all program resources.

SHOW
Specifies the program output fields to be returned. The program name is always returned, along with the name of the IMS that created the output, the region type, and the completion code. The filters supported with the SHOW keyword are:

ALL
Returns all information about the program itself. Other SHOW keywords can be specified to return information about resources related to the program.

BMPTYPE
BMP type option. Specifies whether the program executes in a BMP type region or not. A BMP type region might be a BMP region or a JBP region.

PSBs scheduled by DB2 stored procedures, by programs running under WebSphere® Application Server, and by other users of the ODBA interface may be defined with BMPTYPE Y or N.

DB The databases referenced by the PSB associated with this program. If the PSB intent list is not resident, the ACBLIB is searched to obtain the database information.

The QRY PGM SHOW(DB) command will not show the database names for which the dynamic (DOPT) PSB has intent.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(DB) individually.

DEFNTYPE
Definition type that the descriptor or resource was defined with.

DOPT
Dynamic option.

FP Fast Path option.

GPSB
Generated PSB option.

LANG
Language interface of the application program. The language interface is displayed only for programs defined as GPSB(Y) and programs defined as DOPT(Y) LANG(JAVA).

MODEL
Model name and model type used to create this resource. If the descriptor or resource is created with one or more of the attributes defined and no model specified, the model name and model type is the default descriptor. The model name and model type are blank for IMS-defined resources and descriptors and queue-only transactions created by the DFSINSX0 exit. The CREATE command specified without the LIKE keyword creates a descriptor or resource using the default descriptor as a model. The default

descriptor is either the IMS descriptor DFSDSPG1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor or resource using a model. The descriptor or resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor or resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

RESIDENT
Resident option, which indicates whether the PSB is accessed in local storage. The local runtime value for the resident option is shown. The resident option definition is also shown, if it is different from the runtime value. The RESIDENT(Y) option takes effect at the next IMS restart, unless an error is encountered such as no PSB in the ACBLIB for the program, or if the program was created or updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart.

RTC
routing code descriptors associated with this program.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(RTC) individually.

SCHDTYPE
Scheduling type, which indicates whether this application program can be scheduled into more than one message region or batch message region simultaneously.

STATUS
Program status. For a description of the possible program status returned, see the STATUS keyword under the Output fields table.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

TRAN
Transactions associated with this program.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(TRAN) individually.

TRANSTAT
Transaction level statistics option.

WORK

Work in progress for the program specified on NAME parameter and its associated resources. The QRY PGM SHOW(WORK) command can be issued before a DELETE, IMPORT or UPDATE command to check for any work in progress for the specified program and any of its associated resources. Any work in progress might cause the subsequent DELETE, IMPORT or UPDATE commands to fail. The QRY PGM SHOW(WORK) command returns the work status for the program specified. If no work is in progress for the specified resource, a response line is returned with a work status of blanks.

Notes:

1. SHOW(WORK) specified with NAME(*) might have a performance impact on the processing of the command.
2. You cannot specify this filter with other SHOW filters; you must specify SHOW(WORK) individually.
3. The QRY PGM SHOW(WORK) command is not valid on an XRF alternate.

STATUS()

Selects programs for display that possess at least one of the specified program status. This selection allows for additional filtering by program status. The program status is returned as output, even if the SHOW(STATUS) was not specified.

DB-NOTAVL
A database used by this program is not available, either because it is not defined, or because it is not authorized.

IOPREV
A BMP, IFP, or JBP program cannot complete scheduling, because I/O prevention has not completed. Further I/O requests to data sets are inhibited.

LOCK
Sets the STATUS() filter to return information about programs that are locked.

NOTINIT
Sets the STATUS() filter to return information about programs that are not initialized and therefore cannot be used.

STOSCHD
Sets the STATUS() filter to return information about programs for which program scheduling is stopped.

TRACE
Sets the STATUS() filter to return information about programs that are being traced.

Usage notes

Program resources combined with transactions define the scheduling and resource requirements for an application program. The program resource describes an application program that operates in a message processing region, Fast Path message-driven program region, batch processing region, batch message processing region, or CCTL threads.

This command can be issued only through the Operations Manager API. This command applies to DB/DC, DBCTL and DCCTL systems. This command is allowed on XRF alternate and RSR tracker systems.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY PGM command and the type-1 IMS commands that perform similar functions.

Table 112. Type-1 equivalents for the QUERY PGM command

QUERY PGM command	Similar IMS type-1 command
QUERY PGM SHOW(ALL)	/DISPLAY PROGRAM, /DISPLAY STATUS PROGRAM
QUERY PGM SHOW(DB)	/DISPLAY PSB
QUERY PGM SHOW(RTC)	/DISPLAY PSB
QUERY PGM SHOW(TRAN)	/DISPLAY PROGRAM
QUERY PGM SHOW(WORK)	/DISPLAY MODIFY ALL

Output fields

The following table shows the QUERY PGM output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 113. Output fields for the QUERY PGM command

Short label	Long label	Keyword	Scope	Meaning
BMPT	BMPType	BMPTYPE	LCL	BMP type. **N** The program does not execute in a BMP type region. It might execute in an IMS TM MPP, JMP or IFP region, or it might use either the ODBA interface or the DRA interface. Use this specification for programs that run in IMS TM MPP, JMP, and IFP regions, or PSBs scheduled by CICS® programs using DBCTL and other users of the DRA interface. This is the default. **Y** The program executes in a BMP type region. It might execute in an IMS BMP region or a JBP region. Any associated transactions are assigned normal and limit priority values of zero.
CC	CC	N/A	LCL	Completion code.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DB	DBName	DB	LCL	Database referenced by the program.
DFNT	DefnType	DEFNTYPE	LCL	Definition type, which can be one of the following: **CREATE** Defined by a CREATE command. **DFSINSX0** Defined by user exit DFSINSX0. The program can only be exported if the export option was set. **IMPORT** Defined by an IMPORT command. **IMS** Defined by IMS. DBF#FPU0 is a program created by IMS for the Fast Path Utility. **MODBLKS** Defined by system definition in the MODBLKS data set. The definition type changes from MODBLKS to UPDATE, if an UPDATE PGM command is issued to change a MODBLKS-defined program's attributes. **UPDATE** Defined by system definition in the MODBLKS data set, but changed into a dynamic resource by an UPDATE command.
DOPT	DOPT	DOPT	LCL	Dynamic option (Y) or not (N). **N** The PSB associated with this application program is not located dynamically. The PSB must exist in ACBLIB, otherwise the program is set to a NOTINIT-*xx-reason* status and cannot be scheduled. **Y** The PSB associated with this program is located dynamically. Each time the program associated with this PSB is scheduled, the latest copy of the PSB is loaded from ACBLIB. The PSB does not need to be in any data set defined for ACBLIB until it is actually required to process a transaction. A new version of the PSB can be defined in ACBLIB and is picked up the next time the PSB is scheduled. DOPT PSBs referencing DBDs that are missing from ACBLIB cannot be scheduled. When the program terminates, the PSB is deleted from the PSB pool.

Table 113. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
FP	FP	FP	LCL	Fast Path exclusive program (E) or not (N). **E** The program is a Fast Path-exclusive application program. **N** The program is not a Fast Path application program.
GPSB	GPSB	GPSB	LCL	Generated PSB generated by IMS (Y) or not (N). **N** The PSB associated with the program is not generated by IMS. The PSB must exist in ACBLIB, otherwise the program is set to a NOTINIT-*xx-reason* status and cannot be scheduled. **Y** The PSB associated with the program resource is generated by IMS. It is not loaded from ACBLIB. The scheduling process of all environments generates a PSB containing an I/O PCB and an alternate modifiable PCB. You do not need to perform the PSBGEN and ACBGEN, thus eliminating I/O to the ACBLIB. The generated PSB contains an I/O PCB named IOPCBbbb and a modifiable, alternate PCB named TPPCB1bb. With an alternate modifiable PCB, an application can use the CHNG call to change the output destination and send output to a destination other than the input destination.
LANG	PgmLang	LANG	LCL	Language interface. **ASM/CBL** Assembler or COBOL **JAVA** Java™ (can only run in a Java dependent region) **PASCAL** PASCAL **PLI** PL/I
LRSDNT	LRsdnt	RESIDENT	LCL	Local runtime value of the resident option. Indicates whether the program PSB resides in local storage. If a program is defined as resident but encounters an error during IMS restart, such as no PSB in ACBLIB for the program, this value is N. This value is not returned for program descriptors.

Table 113. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
LSTT	LcLStat	STATUS	LCL	Local application program status. **DB-NOTAVL** A database used by this program is not available, either because it is not defined, or because it is not authorized. **IOPREV** A BMP program containing GSAM cannot complete scheduling because I/O prevention has not completed. Further I/O requests to data sets are inhibited. **LOCK** Program is locked. **NOTINIT-***xx-reason* Programs that are not initialized and therefore cannot be used. NOTINIT is displayed in the format NOTINIT-*xx-reason*. *xx* is the code that identifies the unique location in one module where this reason code is set, which is used by IBM for diagnostics. If the suggested action is to call IBM, the *xx* value will help IBM identify exactly where this resource was marked bad. DFSPDIR MACRO defines each reason code that might be set in the program bad reason code (field PDIRBADR) and identifies the module that sets it. NOTINIT-00 indicates that the reason is unknown. Action: 1. *reason* explains the reason code *xx* in abbreviated text format up to 13 characters. **STOSCHD** Program scheduling is stopped. **TRACE** Program is being traced.
MBR	MbrName	N/A	LCL	IMSplex member that build the output line.
MDLN	ModelName	MODEL	LCL	Model name. Name of the resource used as a model to create this resource. DFSDSPG1 is the IMS descriptor name for programs.
MDLT	ModelType	MODEL	LCL	Model type, either RSC or DESC. RSC means the resource was created using another resource as a model. DESC means the resource was created using a descriptor as a model.
PGM	PgmName	PGM	LCL	Program name that references the database.
RGNT	RgnType	N/A	LCL	Region type in which program can run. Some programs can run in additional region types. For example, a program defined with a program type of MSG can run in a BMP under certain conditions. BMP indicates a batch message processing region. FPU indicates a Fast Path utility region. IFP indicates a Fast Path message processing region. JBP indicates a Java batch message processing region. JMP indicates a Java message processing region. MPP indicates an MPP processing region.

Table 113. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
RSDNT	Rsdnt	RESIDENT	LCL	Resident option. Indicates whether the program PSB is to reside in local storage at the next IMS restart. This value is only returned if the resident option definition is different from the resident option local runtime value. **N** The PSB associated with the named application program resource is not resident in storage. The PSB is loaded at scheduling time. **Y** The PSB associated with the named application program resource is resident in storage. A resident PSB is accessed in local storage, which avoids I/O to the ACBLIB. The PSB associated with the program is made resident in storage at IMS cold start or restart. IMS loads the PSB and initializes it.
RTC	Rtcode	RTC	LCL	Routing code associated with the program.
SCHD	SchdType	SCHDTYPE	LCL	Schedule type. **PARALLEL** The application program can be scheduled into more than one message region or batch message region simultaneously. **SERIAL** The application program can only be scheduled in one region at a time.
TLS	TranStat	TRANSTAT	LCL	Transaction level statistics logged (Y) or not (N). **N** Transaction level statistics logging is not active. **Y** Transaction level statistics logging is active.
TMAC	TimeAccess	TIMESTAMP	LCL	The time that the resource was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. For a program resource, the following actions update the last access time: • Program is scheduled. • CREATE command or DFSINSX0 exit references the resource as a model.
TMCR	TimeCreate	TIMESTAMP	LCL	The time that the resource was created with a CREATE PGM command, an IMPORT command that creates the program, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMIM	TimeImport	TIMESTAMP	LCL	The time that the resource was last imported, if applicable. The import time is retained across warm start and emergency restart.
TMUP	TimeUpdate	TIMESTAMP	LCL	The time that the resource was updated as a result of an UPDATE PGM command that changes a runtime resource definition attribute. This includes attributes set by the SET keyword. The update time is retained across warm start, emergency restart, EXPORT and IMPORT.
TRAN	Tran	TRAN	LCL	Transaction associated with the program.

Table 113. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
WRK	Work	WORK	LCL	Work is in progress for the program or one of its associated resources. The work in progress can be one of the following: ANOTHER CMD IN PROGRESS Another command (such as DELETE or UPDATE) to delete or update the program is already in progress, or a command to delete or update a transaction referencing the program to be updated is in progress. SCHEDULED Program is scheduled.

Table 114. Reason information for NOTINIT-xx-reason status

Reason	Meaning
ACBLIBREAD	I/O error reading ACBLIB.
ALIAS	Alias name error.
BLDL	BLDL miscellaneous error trying to build ACBLIB directory.
DBFINTP0	Module DBFINTP0 returns nonzero return code.
DBNOTINIT	Program references a database that is not initialized and has a NOTINIT status. Action: 2.
DMBINCOMPTBL	DMB incompatibility.
DMBNUM	No DDIR control block for database with this DMB number exists. Action: 1.
DMBPOOL	DMB pool shortage.
EOD	EOD marker found before DMB.
FPDB	The database is a Fast MSDB or DEDB that is defined in a non-Fast Path (FP=N) system. The database cannot be used.
FPMISMATCH	The Fast Path database DDIR control block does not match with the non-Fast Path PCB.
FPRESTART	A Fast Path error occurred during restart. Action: 1.
INTLISTDB	Intent list contains unknown database. Action: 5.
INTLISTLEN	Intent list read length is invalid.
INTLISTLEN0	Intent list read length is zero.
INTLISTREAD	Intent list read error.
KSDSESDS	Missing KSDS/ESDS.
NAMEMISMATCH	PSB/PDIR names do not match.
NOACB	No block found in ACBLIB.
NOAMP	DMB not pointing to access method prefix block (AMP).
NOAMPOFLWDCB	Access method prefix block (AMP) missing overflow DCB.
NODB	A database this program references is not defined. No database DDIR control block exists. Action: 5. See message DFS563I for the name of the database that is not defined.
NODMB	No DMB exists in ACBLIB for the database. Action: 2, 5, or both.
NOPSB	No PSB exists in ACBLIB for the program. Action: 3.
NOTDMB	Not a DMB.

Table 114. Reason information for NOTINIT-xx-reason status (continued)

Reason	Meaning
NOTPSB	Not a PSB. A DMB by the same name as the program is defined in ACBLIB, instead of a PSB. If this resource should be a database, create the database with a CREATE DB command. If this resource should be a program, perform a DBDGEN, PSBGEN, ACBGEN and ACBLIB online change to define this resource as a PSB instead of a DMB. Action: 4.
PROCOPTL	PCB specifies PROCOPT=L for online.
PSBLEN	Bad PSB length.
PSBLEVEL	The IMS release level at which this PSB is generated using ACBGEN does not match the IMS release level of this IMS. Perform a PSBGEN, ACBGEN, and ACBLIB online change as needed to generate this PSB at the correct IMS release level. Action: 4.
PSBLOGICAL	PSB is missing a logical relationship database. Action: 4.
PSBPOOL	PSB pool shortage.
SIZEMISMATCH	Size mismatch.
WRONGPSB	Wrong PSB.

Note: Actions that can be taken to initialize the program are:
1. Call IBM.
2. Perform an ACBLIB online change to add the DMB to ACBLIB.
3. Perform an ACBLIB online change to add the PSB to ACBLIB.
4. Perform an ACBLIB online change to correct PSBs, DMBs, or both.
5. Perform a MODBLKS online change or issue a CREATE DB command to create the database; then issue an UPDATE PGM START(SCHD) command to reset the NOTINIT status.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 115. Return and reason codes for the QUERY PGM command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000004'	X'00001010'	No resources were found to be returned. The resource name(s) specified might be invalid, or there were no resources that match the filter specified, or there were no resources that had work to display for the SHOW(WORK) specified.
X'00000008'	X'00002040'	Invalid filter or filter combination. An invalid filter might be an invalid parameter specified with the SHOW keyword or STATUS keyword. An invalid filter combination might be more than one filter specified (SHOW, STATUS), or more than one SHOW filter specified that includes DB, RTC, TRAN or WORK.

Table 115. Return and reason codes for the QUERY PGM command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'0000000C'	X'00003004'	Command was successful for none of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004016'	The command failed because it is not valid in a DCCTL environment.
X'00000010'	X'00004024'	No Fast Path defined. SHOW(RTC) is not permitted.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 116. Completion codes for the QUERY PGM command

Completion code	Completion code text	Meaning
0		Command completed successfully for program.
10	NO RESOURCES FOUND	Program name is invalid, or the wildcard parameter specified does not match any resource names.
28	NO DMB LOADED	The DMB that is associated with the program does not exist in ACBLIB.

Examples

The following are examples of the QUERY PGM command:

Example 1 for QUERY PGM command

TSO SPOC input:

```
QUERY PGM NAME(PGM0000%,APOL1,BMP255,DBF*0,DFSIVP67,AUTPSB7,JAVPSB1,JAVTESTJ,
DCSQL7A,DCSQL6C,JVMJBP1) SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
PgmName  MbrName    CC RgnType BMPType FP DOPT GPSB LRsdnt TranStat PgmLang  Sch
APOL1    IMS1        0 MPP     N       N  N    N    N      N                 SER
AUTPSB7  IMS1        0 JBP     Y       N  N    N    N      N                 SER
BMP255   IMS1        0 BMP     Y       N  N    N    N      N                 PAR
DBF#FPU0 IMS1        0 FPU     N       E  Y    N    N      N                 PAR
DCSQL6C  IMS1        0 MPP     N       N  N    Y    N      N        PLI      SER
DCSQL7A  IMS1        0 MPP     N       N  N    Y    N      N        ASM/CBL  SER
```

```
DFSIVP67 IMS1        0 JBP     Y       N  N    N    N      N                SER
JAVPSB1  IMS1        0 IFP     N       E  N    N    Y      N                SER
JAVTESTJ IMS1        0 JMP     N       N  N    N    N      N                PAR
JVMJBP1  IMS1        0 JBP     Y       N  N    Y    N      N        JAVA    SER
PGM00001 IMS1        0 BMP     Y       N  N    N    N      N                PAR
PGM00002 IMS1        0 IFP     N       E  Y    N    N      N                SER
PGM00003 IMS1        0 MPP     N       N  N    Y    N      N        PASCAL  PAR
PGM00004 IMS1        0 BMP     Y       N  Y    N    N      N                PAR
(scrolled right to screen 2)
PgmName  MbrName  RgnType hdType LclStat          ModelName ModelType TimeCreate
APOL1    IMS1     MPP     RIAL   STOSCHD                              2006.286 0
AUTPSB7  IMS1     JBP     RIAL                                        2006.286 0
BMP255   IMS1     BMP     RALLEL                                      2006.286 0
DBF#FPU0 IMS1     FPU     RALLEL                                      2006.286 0
DCSQL6C  IMS1     MPP     RIAL                                        2006.286 0
DCSQL7A  IMS1     MPP     RIAL                                        2006.286 0
DFSIVP67 IMS1     JBP     RIAL                                        2006.286 0
JAVPSB1  IMS1     IFP     RIAL                                        2006.286 0
JAVTESTJ IMS1     JMP     RALLEL                                      2006.286 0
JVMJBP1  IMS1     JBP     RIAL                                        2006.286 0
PGM00001 IMS1     BMP     RALLEL NOTINIT-26-NOPSB DFSDSPG1  DESC      2006.286 1
PGM00002 IMS1     IFP     RIAL                    FPEDESC   DESC      2006.286 1
PGM00003 IMS1     MPP     RALLEL                  DFSDSPG1  DESC      2006.286 1
PGM00004 IMS1     BMP     RIAL                    BMP011    RSC       2006.286 1
(scrolled right to screen 3)
PgmName  MbrName  RgnType TimeCreate TimeUpdate           TimeAccess           T
APOL1    IMS1     MPP     9:14:53.37
AUTPSB7  IMS1     JBP     9:14:53.37
BMP255   IMS1     BMP     9:14:53.37
DBF#FPU0 IMS1     FPU     9:14:53.37
DCSQL6C  IMS1     MPP     9:14:53.37
DCSQL7A  IMS1     MPP     9:14:53.37
DFSIVP67 IMS1     JBP     9:14:53.37
JAVPSB1  IMS1     IFP     9:14:53.37
JAVTESTJ IMS1     JMP     9:14:53.37
JVMJBP1  IMS1     JBP     9:14:53.37
PGM00001 IMS1     BMP     0:40:53.83
PGM00002 IMS1     IFP     0:42:06.73
PGM00003 IMS1     MPP     1:00:10.11
PGM00004 IMS1     BMP     1:03:00.87
(scrolled right to screen 4)
PgmName  MbrName  RgnType TimeAccess           TimeImport           DefnType
APOL1    IMS1     MPP                                              MODBLKS
AUTPSB7  IMS1     JBP                                              MODBLKS
BMP255   IMS1     BMP                                              MODBLKS
DBF#FPU0 IMS1     FPU                                              IMS
DCSQL6C  IMS1     MPP                                              MODBLKS
DCSQL7A  IMS1     MPP                                              MODBLKS
DFSIVP67 IMS1     JBP                                              MODBLKS
JAVPSB1  IMS1     IFP                                              MODBLKS
JAVTESTJ IMS1     JMP                                              MODBLKS
JVMJBP1  IMS1     JBP                                              MODBLKS
PGM00001 IMS1     BMP                                              CREATE
PGM00002 IMS1     IFP                                              CREATE
PGM00003 IMS1     MPP                                              CREATE
PGM00004 IMS1     BMP                                              CREATE
```

OM API input:

```
CMD(QUERY PGM NAME(PGM0000%,APOL1,BMP255,DBF*0,DFSIVP67,AUTPSB7,JAVPSB1,JAVTESTJ,
DCSQL7A,DCSQL6C,JVMJBP1) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
```

```
<xmlvsn>20  </xmlvsn>
<statime>2006.286 18:09:20.978421</statime>
<stotime>2006.286 18:09:20.979045</stotime>
<staseq>BF8C578B1C1F548E</staseq>
 <stoseq>BF8C578B1C46510E</stoseq>
 <rqsttkn1>USRT011 10110920</rqsttkn1>
 <rc>00000000</rc>
 <rsn>00000000</rsn>
 </ctl>
 <cmd>
 <master>IMS1    </master>
 <userid>USRT011 </userid>
 <verb>QRY </verb>
 <kwd>PGM             </kwd>
 <input>QRY PGM
  NAME(PGM0000%,APOL1,BMP255,DBF*0,DFSIVP67,AUTPSB7,JAVPSB1,JAVTESTJ,
 DCSQL7A,DCSQL6C,JVMJBP1) SHOW(ALL) </input>
 </cmd>
 <cmdrsphdr>
 <hdr slbl="PGM" llbl="PgmName" scope="LCL" sort="a" key="1" scroll="no"
  len="8" dtype="CHAR" align="left" />
 <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
  len="8" dtype="CHAR" align="left" />
 <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
  len="4" dtype="INT" align="right" skipb="no" />
 <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
  scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="RGNT" llbl="RgnType" scope="LCL" sort="n" key="0"
 scroll="no" len="7" dtype="CHAR" align="left" />
<hdr slbl="BMPT" llbl="BMPType" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="CHAR" align="left" />
<hdr slbl="FP" llbl="FP" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="DOPT" llbl="DOPT" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="GPSB" llbl="GPSB" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="RSDNT" llbl="Rsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="LRSDNT" llbl="LRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="TLS" llbl="TranStat" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="LANG" llbl="PgmLang" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="SCHD" llbl="SchdType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" />
<hdr slbl="MDLN" llbl="ModelName" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 <hdr slbl="MDLT" llbl="ModelType" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 <hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMUP" llbl="TimeUpdate" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMAC" llbl="TimeAccess" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMIM" llbl="TimeImport" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="DFNT" llbl="DefnType" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 </cmdrsphdr>
 <cmdrspdata>
 <rsp>PGM(PGM00001) MBR(IMS1) CC(   0) RGNT(BMP) BMPT(Y) FP(N) DOPT(N)
  GPSB(N) LRSDNT(N) TLS(N) SCHD(PARALLEL) LSTT(NOTINIT-26-NOPSB)
```

```
| MDLT(DESC) MDLN(DFSDSPG1) TMCR(2006.286 10:40:53.83) DFNT(CREATE)
| </rsp>
| <rsp>PGM(PGM00002) MBR(IMS1) CC(   0) RGNT(IFP) BMPT(N) FP(E) DOPT(Y)
| GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL) LSTT() MDLT(DESC) MDLN(FPEDES
| TMCR(2006.286 10:42:06.73) DFNT(CREATE) </rsp>
|<rsp>PGM(PGM00003) MBR(IMS1) CC(   0) RGNT(MPP) BMPT(N) FP(N) DOPT(N)
| GPSB(Y) LRSDNT(N) TLS(N) LANG(PASCAL) SCHD(PARALLEL) LSTT() MDLT(DESC)
| MDLN(DFSDSPG1) TMCR(2006.286 11:00:10.11) DFNT(CREATE) </rsp>
| <rsp>PGM(PGM00004) MBR(IMS1) CC(   0) RGNT(BMP) BMPT(Y) FP(N) DOPT(Y)
| GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL) LSTT() MDLT(RSC) MDLN(BMP011  )
| TMCR(2006.286 11:03:00.87) DFNT(CREATE) </rsp>
| <rsp>PGM(APOL1   ) MBR(IMS1) CC(   0) RGNT(MPP) BMPT(N) FP(N) DOPT(N)
| GPSB(N) LRSDNT(N) TLS(N) LANG() SCHD(SERIAL) LSTT(STOSCHD)
| TMCR(2006.286 09:14:53.37) DFNT(MODBLKS) </rsp>
| <rsp>PGM(BMP255  ) MBR(IMS1) CC(   0) RGNT(BMP) BMPT(Y) FP(N) DOPT(N)
| GPSB(N) LRSDNT(N) TLS(N) SCHD(PARALLEL) LSTT() TMCR(2006.286
| 09:14:53.37) DFNT(MODBLKS) </rsp>
| <rsp>PGM(DBF#FPU0) MBR(IMS1) CC(   0) RGNT(FPU) BMPT(N) FP(E) DOPT(Y)
| GPSB(N) LRSDNT(N) TLS(N) SCHD(PARALLEL) LSTT() TMCR(2006.286
| 09:14:53.37) DFNT(IMS) </rsp>
| <rsp>PGM(DFSIVP67) MBR(IMS1) CC(   0) RGNT(JBP) BMPT(Y) FP(N) DOPT(N)
| GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL) LSTT() TMCR(2006.286
| 09:14:53.37) DFNT(MODBLKS) </rsp>
| <rsp>PGM(AUTPSB7 ) MBR(IMS1) CC(   0) RGNT(JBP) BMPT(Y) FP(N) DOPT(N)
| GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL) LSTT() TMCR(2006.286
| 09:14:53.37) DFNT(MODBLKS) </rsp>
| <rsp>PGM(JAVPSB1 ) MBR(IMS1) CC(   0) RGNT(IFP) BMPT(N) FP(E) DOPT(N
| GPSB(N) LRSDNT(Y) TLS(N) SCHD(SERIAL) LSTT() TMCR(2006.286
|09:14:53.37) DFNT(MODBLKS) </rsp>
| <rsp>PGM(JAVTESTJ) MBR(IMS1) CC(   0) RGNT(JMP) BMPT(N) FP(N) DOPT(N)
| GPSB(N) LRSDNT(N) TLS(N) SCHD(PARALLEL) LSTT() TMCR(2006.286
| 09:14:53.37) DFNT(MODBLKS) </rsp>
| <rsp>PGM(DCSQL7A ) MBR(IMS1) CC(   0) RGNT(MPP) BMPT(N) FP(N) DOPT(N)
| GPSB(Y) LRSDNT(N) TLS(N) LANG(ASM/CBL) SCHD(SERIAL) LSTT()
| TMCR(2006.286 09:14:53.37) DFNT(MODBLKS) </rsp>
| <rsp>PGM(DCSQL6C ) MBR(IMS1) CC(   0) RGNT(MPP) BMPT(N) FP(N) DOPT(N)
| GPSB(Y) LRSDNT(N) TLS(N) LANG(PLI) SCHD(SERIAL) LSTT() TMCR(2006.286
| 09:14:53.37) DFNT(MODBLKS) </rsp>
| <rsp>PGM(JVMJBP1 ) MBR(IMS1) CC(   0) RGNT(JBP) BMPT(Y) FP(N) DOPT(N)
| GPSB(Y) LRSDNT(N) TLS(N) LANG(JAVA) SCHD(SERIAL) LSTT() TMCR(2006.286
| 09:14:53.37) DFNT(MODBLKS) </rsp>
| </cmdrspdata>
| </imsout>
```

Explanation: Lots of different types of programs are displayed. SHOW(ALL) is specified to show all of the possible output fields. All of the program output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The program name, member name that built the line of output, and region type in which the program can run are displayed on every screen. Program names starting with PGM0000 were dynamically created with CREATE PGM commands. DBF#FPU0 is the Fast Path Utility. JAVTESTJ is a Java message processing program. PGM00001 has no PSB in ACBLIB, so it shows a local status of NOTINIT-26-NOPSB and cannot be scheduled. PGM00003 is a generated PSB GPSB(Y), so it does not need a PSB in ACBLIB. PGM00002 and PGM00004 are dynamic option DOPT(Y) programs, so IMS will not detect that there is no PSB for them in ACBLIB and mark their status as NOTINIT until the first time they are scheduled. A few GPSBs are displayed with different language definitions. The language attribute applies to GPSBs and DOPT PSBs defined as Java programs.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY MODIFY command (Commands)

/DISPLAY PGM command (Commands)

/DISPLAY PSB command (Commands)

/DISPLAY STATUS command (Commands)

Command keywords and their synonyms (Commands)

QUERY PGMDESC command

Use the QUERY PGMDESC command to query information about program descriptors. A descriptor is a model that can be used to create descriptors or resources.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 250
- "Usage notes" on page 251
- "Output fields" on page 252
- "Return, reason, and completion codes" on page 255
- "Examples" on page 256

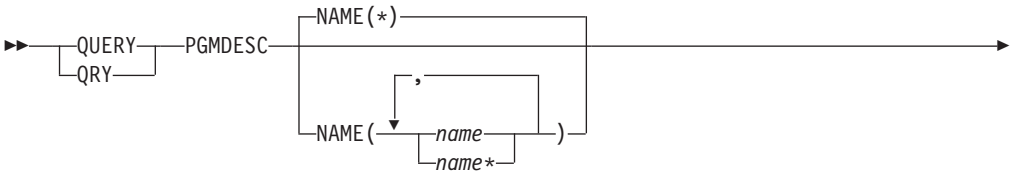
Environment

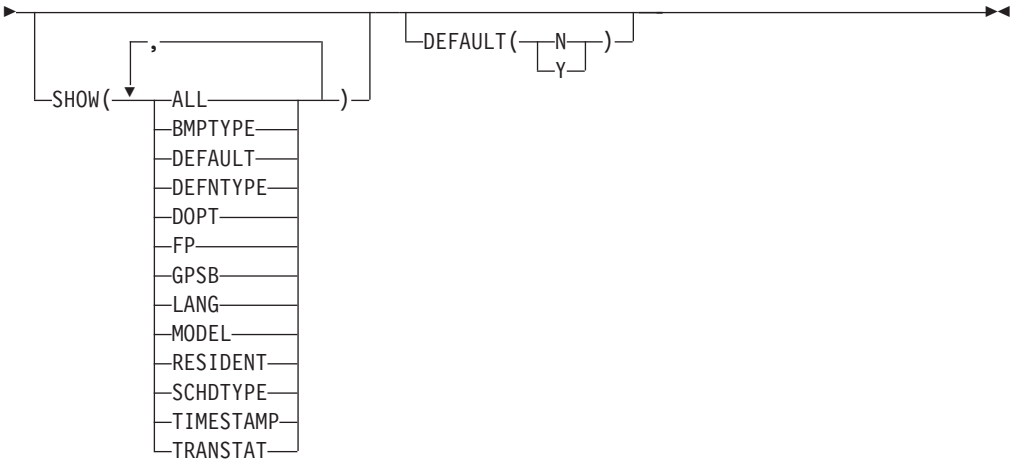
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 117. Valid environments for the QUERY PGMDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY PGMDESC	X	X	X
NAME	X	X	X
SHOW	X	X	X
STATUS	X	X	X
DEFAULT	X	X	X

Syntax





Keywords

The following keywords are valid for the QUERY PGMDESC command:

NAME
Specifies the 1-8 character name of the program descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*) which returns all program descriptors.

DEFAULT()
Specifies which descriptor or descriptors to display.

N Displays all the program descriptors that are not the default.

Y Displays the default program descriptor.

SHOW
Specifies the program descriptor output fields to be returned. The program descriptor name is always returned, along with the name of the IMS that created the output, the region type, and the completion code. The filters that are supported with the SHOW keyword are:

ALL
Returns all information about the program descriptor itself.

BMPTYPE
BMP type option. Specifies whether the program runs in a BMP type region or not. A BMP type region might be a BMP region or a JBP region.

PSBs scheduled by DB2 stored procedures, by programs running under WebSphere Application Server, and by other users of the ODBA interface can be defined with BMPTYPE Y or N.

DEFAULT
Default descriptor option.

DEFNTYPE
Definition type. This is how the descriptor was defined.

DOPT
Dynamic option.

FP Fast Path option.

GPSB
Generated PSB option.

LANG
Language interface of the application program for a GPSB.

MODEL
Model name and model type used to create this descriptor. If the descriptor is created with one or more attributes defined and no model specified, the model name and model type is the default descriptor. The model name and model type are blank for IMS-defined descriptors. The CREATE command specified without the LIKE keyword creates a descriptor using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSPG1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor using a model. The descriptor is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify descriptors that were created with the same model. The model name and model type of a descriptor are exported and imported. The IMPORT command does not use the model name and model type when creating a descriptor.

RESIDENT
Resident option.

SCHDTYPE
Scheduling type, which indicates whether this application program can be scheduled into more than one message region or batch message region simultaneously.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

TRANSTAT
Transaction level statistics option.

Usage notes

This command can be issued only through the Operations Manager API. This command applies to DB/DC, DBCTL and DCCTL systems. This command is allowed on XRF alternate and RSR tracker systems. The QUERY PGMDESC command is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

Many of the attributes defined with the program descriptor become meaningful only for programs that are created using the program descriptor. Program descriptors are never scheduled; only programs created using a program descriptor are scheduled.

Output fields

The following table shows the QUERY PGMDESC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 118. Output fields for the QUERY PGMDESC command

Short label	Keyword	Scope	Meaning
BMPT	BMPTYPE	LCL	BMP type. **N** The program does not execute in a BMP type region. It might execute in an IMS TM MPP, JMP or IFP region, or it might use either the ODBA interface or the DRA interface. Use this specification for programs that run in IMS TM MPP, JMP, and IFP regions, or PSBs scheduled by CICS programs using DBCTL and other users of the DRA interface. This is the default. **Y** The program executes in a BMP type region. It might execute in an IMS BMP region or a JBP region. Any associated transactions are assigned normal and limit priority values of zero.
CC	N/A	LCL	Completion code.
CCTXT	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DESC	PGMDESC	LCL	Program descriptor name.
DFLT	DEFAULT	LCL	Default descriptor (Y) or not (N). **N** The descriptor is not the default. **Y** The descriptor is the default. When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a program descriptor called DFSDSPG1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor.

Table 118. Output fields for the QUERY PGMDESC command (continued)

Short label	Keyword	Scope	Meaning
DFNT	DEFNTYPE	LCL	Definition type, which can be one of the following: **CREATE** Defined by a CREATE command. The DEFNTYPE is not changed if the descriptor is updated with an UPDATE command. **IMPORT** Defined by an IMPORT command. The DEFNTYPE is not changed if the descriptor is updated with an UPDATE command. **IMS** Defined by IMS. DFSDSPG1 is an IMS-defined program descriptor containing the default program values.
DOPT	DOPT	LCL	Dynamic option (Y) or not (N). **N** The PSB associated with this application program is not located dynamically. **Y** The PSB associated with this program is located dynamically. Each time the program associated with this PSB is scheduled, the latest copy of the PSB is loaded from ACBLIB. The PSB does not need to be in any data set defined for ACBLIB until it is actually required to process a transaction. A new version of the PSB can be defined in ACBLIB and is picked up the next time the PSB is scheduled. DOPT PSBs referencing DBDs that are missing from ACBLIB cannot be scheduled. When the program terminates, the PSB is deleted from the PSB pool.
FP	FP	LCL	Fast Path exclusive program (E) or not (N). **E** The program is a Fast Path-exclusive application program. **N** The program is not a Fast Path application program.
GPSB	GPSB	LCL	Generated PSB generated by IMS (Y) or not (N). **N** The PSB associated with the program is not generated by IMS. **Y** The PSB associated with the program descriptor is generated by IMS. It is not loaded from ACBLIB. The scheduling process of all environments generates a PSB containing an I/O PCB and an alternate modifiable PCB. You do not need to perform the PSBGEN and ACBGEN, thus eliminating I/O to the ACBLIB. The generated PSB contains an I/O PCB named IOPCBbbb and a modifiable, alternate PCB named TPPCB1bb. With an alternate modifiable PCB, an application can use the CHNG call to change the output destination and send output to a destination other than the input destination.

Table 118. Output fields for the QUERY PGMDESC command (continued)

Short label	Keyword	Scope	Meaning
LANG	LANG	LCL	Language interface. **ASM/CBL** Assembler or COBOL **JAVA** JAVA (can only run in a Java dependent region). **PASCAL** PASCAL **PLI** PL/I
MBR	N/A	LCL	IMSplex member that build the output line.
MDLN	MODEL	LCL	Model name. Name of the descriptor used as a model to create this resource. DFSDSPG1 is the IMS descriptor name for programs.
MDLT	MODEL	LCL	Model type, either RSC or DESC. RSC means the descriptor was created using another resource as a model. DESC means the resource was created using a descriptor as a model.
RGNT	N/A	LCL	Region type in which program can run. Some programs can run in additional region types. For example, a program defined with a program type of MSG can run in a BMP under certain conditions. BMP indicates a batch message processing region. FPU indicates a Fast Path utility region. IFP indicates a Fast Path message processing region. JBP indicates a JAVA batch message processing region. JMP indicates a JAVA message processing region. MPP indicates an MPP processing region.
LRSDNT	RESIDENT	LCL	Local runtime value of the resident option. Indicates whether the program PSB resides in local storage. If a program is defined as resident but encounters an error during IMS restart, such as no PSB in ACBLIB for the program, this value is N. This value is not returned for program descriptors.
RSDNT	RESIDENT	LCL	Resident option. Indicates whether the program PSB is to reside in local storage at the next IMS restart. This value is only returned if the resident option definition is different from the resident option local runtime value. **N** The PSB associated with the named application program descriptor is not resident in storage. The PSB is loaded at scheduling time. **Y** The PSB associated with the named application program descriptor is resident in storage. When the PSB is accessed, this avoids I/O to the ACBLIB.

Table 118. Output fields for the QUERY PGMDESC command (continued)

Short label	Keyword	Scope	Meaning
SCHD	SCHDTYPE	LCL	Schedule type. **PARALLEL** The application program can be scheduled into more than one message region or batch message region simultaneously. **SERIAL** The application program can only be scheduled in one region at a time.
TLS	TRANSTAT	LCL	Transaction level statistics logged (Y) or not (N). **N** Transaction level statistics logging is not active. **Y** Transaction level statistics logging is active.
TMAC	TIMESTAMP	LCL	The time that the descriptor was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. For a program descriptor, when the CREATE command or DFSINSX0 exit references the descriptor as a model, the last access time is updated.
TMCR	TIMESTAMP	LCL	The time that the descriptor was created. This is the result of a CREATE PGMDESC command, an IMPORT command that creates the descriptor, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMIM	TIMESTAMP	LCL	The time that the descriptor was last imported, if applicable. The import time is retained across warm start and emergency restart.
TMUP	TIMESTAMP	LCL	The time that the descriptor was updated as a result of an UPDATE PGMDESC command that changes a runtime resource definition attribute. This includes attributes set by the SET keyword. The update time is retained across warm start, emergency restart, EXPORT and IMPORT.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 119. Return and reason codes for the QUERY PGMDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000008'	X'00002040'	Invalid filter or filter combination. An invalid filter may be an invalid parameter specified with the DEFAULT keyword.

Table 119. Return and reason codes for the QUERY PGMDESC command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003000'	Command was successful for some descriptors but failed for others. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'0000000C'	X'00003004'	Command was successful for none of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 120. Completion codes for the QUERY PGMDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for program descriptor.
10	NO RESOURCES FOUND	Program descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.

Examples

The following are examples of the QUERY PGMDESC command:

Example 1 for QUERY PGMDESC command

TSO SPOC input:

```
QRY PGMDESC SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
DescName MbrName    CC RgnType BMPType FP DOPT GPSB Rsdnt TranStat PgmLang  SchdType
DFSDSPG1 IMS1        0 MPP     N       N  N    N    N     N                 PARALLEL
DOPTDESC IMS1        0 BMP     Y       N  Y    N    N     N                 SERIAL
FPEDESC  IMS1        0 IFP     N       E  Y    N    N     N                 SERIAL
GPSBDESC IMS1        0 BMP     Y       N  N    Y    N     N        ASM/CBL  SERIAL
JAVADESC IMS1        0 JMP     N       N  Y    N    N     N        JAVA     SERIAL
(scrolled to the right screen 2)
DescName MbrName  RgnType dType ModelName ModelType Dflt TimeCreate
DFSDSPG1 IMS1     MPP     ALLEL                     Y    2006.286 09:14:53.37
DOPTDESC IMS1     BMP     IAL   DFSDSPG1  DESC      N    2006.286 11:33:11.40
FPEDESC  IMS1     IFP     IAL   DFSDSPG1  DESC      N    2006.286 10:41:22.74
GPSBDESC IMS1     BMP     IAL   DFSDSPG1  DESC      N    2006.286 11:32:07.20
JAVADESC IMS1     JMP     IAL   DFSDSPG1  DESC      N    2006.286 11:31:59.06
```

```
(scrolled to the right screen 3)
DescName MbrName  RgnType <TimeUpdate        TimeAccess           TimeImport>
DFSDSPG1 IMS1     MPP                       2006.286 11:33:11.40
DOPTDESC IMS1     BMP
FPEDESC  IMS1     IFP                       2006.286 10:42:06.73
GPSBDESC IMS1     BMP
JAVADESC IMS1     JMP
(scrolled to the right screen 4)
DescName MbrName  RgnType TimeAccess           TimeImport           DefnType
DFSDSPG1 IMS1     MPP     2006.286 11:33:11.40                       IMS
DOPTDESC IMS1     BMP                                                CREATE
FPEDESC  IMS1     IFP     2006.286 10:42:06.73                       CREATE
GPSBDESC IMS1     BMP                                                CREATE
JAVADESC IMS1     JMP
```

OM API input:

```
CMD(QUERY PGMDESC SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.286 18:35:34.337129</statime>
<stotime>2006.286 18:35:34.337631</stotime>
<staseq>BF8C5D6794E69802</staseq>
<stoseq>BF8C5D679505FC42</stoseq>
<rqsttkn1>USRT011 10113534</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>PGMDESC         </kwd>
<input>QRY PGMDESC SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="RGNT" llbl="RgnType" scope="LCL" sort="n" key="0"
scroll="no" len="7" dtype="CHAR" align="left" />
 <hdr slbl="BMPT" llbl="BMPType" scope="LCL" sort="n" key="0"
  scroll="yes" len="7" dtype="CHAR" align="left" />
 <hdr slbl="FP" llbl="FP" scope="LCL" sort="n" key="0" scroll="yes"
  len="1" dtype="CHAR" align="left" />
 <hdr slbl="DOPT" llbl="DOPT" scope="LCL" sort="n" key="0" scroll="yes"
  len="1" dtype="CHAR" align="left" />
 <hdr slbl="GPSB" llbl="GPSB" scope="LCL" sort="n" key="0" scroll="yes"
  len="1" dtype="CHAR" align="left" />
 <hdr slbl="RSDNT" llbl="Rsdnt" scope="LCL" sort="n" key="0"
  scroll="yes" len="1" dtype="CHAR" skipb="yes" align="left" />
 <hdr slbl="TLS" llbl="TranStat" scope="LCL" sort="n" key="0" scroll="y
  es" len="1" dtype="CHAR" align="left" />
 <hdr slbl="LANG" llbl="PgmLang" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 <hdr slbl="SCHD" llbl="SchdType" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
```

```
 <hdr slbl="MDLN" llbl="ModelName" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 <hdr slbl="MDLT" llbl="ModelType" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 <hdr slbl="DFLT" llbl="Dflt" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMUP" llbl="TimeUpdate" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMAC" llbl="TimeAccess" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMIM" llbl="TimeImport" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="DFNT" llbl="DefnType" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 </cmdrsphdr>
 <cmdrspdata>
 <rsp>DESC(DOPTDESC) MBR(IMS1) CC(   0) RGNT(BMP) BMPT(Y) FP(N) DOPT(Y)
  GPSB(N) RSDNT(N) TLS(N) SCHD(SERIAL) MDLT(DESC) MDLN(DFSDSPG1) DFLT(N)
  TMCR(2006.286 11:33:11.40) DFNT(CREATE) </rsp>
 <rsp>DESC(DFSDSPG1) MBR(IMS1) CC(   0) RGNT(MPP) BMPT(N) FP(N) DOPT(N)
  GPSB(N) RSDNT(N) TLS(N) SCHD(PARALLEL) DFLT(Y) TMCR(2006.286
  09:14:53.37) TMAC(2006.286 11:33:11.40) DFNT(IMS) </rsp>
 <rsp>DESC(JAVADESC) MBR(IMS1) CC(   0) RGNT(JMP) BMPT(N) FP(N) DOPT(Y)
  GPSB(N) RSDNT(N) TLS(N) LANG(JAVA) SCHD(SERIAL) MDLT(DESC)
  MDLN(DFSDSPG1) DFLT(N) TMCR(2006.286 11:31:59.06) DFNT(CREATE) </rsp>
 <rsp>DESC(FPEDESC ) MBR(IMS1) CC(   0) RGNT(IFP) BMPT(N) FP(E) DOPT(N)
  GPSB(N) RSDNT(N) TLS(N) SCHD(SERIAL) MDLT(DESC) MDLN(DFSDSPG1) DFLT(N)
TMCR(2006.286 10:41:22.74) TMAC(2006.286 10:42:06.73) DFNT(CREATE)
 </rsp>
 <rsp>DESC(GPSBDESC) MBR(IMS1) CC(   0) RGNT(BMP) BMPT(Y) FP(N) DOPT(N)
  GPSB(Y) RSDNT(N) TLS(N) LANG(ASM/CBL) SCHD(SERIAL) MDLT(DESC)
  MDLN(DFSDSPG1) DFLT(N) TMCR(2006.286 11:32:07.20) DFNT(CREATE) </rsp>
 </cmdrspdata>
 </imsout>
```

Explanation: All program descriptors are returned with all output fields. All of the program descriptor output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The program descriptor name, the member name that built the line of output, and the region type in which the program can run are displayed on every screen. The fields that are blank are not applicable to the specified program descriptor.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY POOL command

Use the QUERY POOL command to display information about the current usage of the buffers that are managed by the Fast Path 64-bit buffer manager or processor storage utilization statistics for the pool type specified.

Subsections:

- "Environment" on page 259
- "Syntax" on page 259
- "Keywords" on page 259

- "Usage notes"
- "Output fields" on page 260
- "Return, reason, and completion codes" on page 262
- "Examples" on page 263

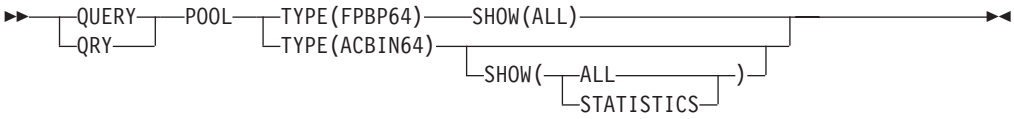
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 121. Valid environments for the QUERY POOL command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY POOL	X	X	X
TYPE	X	X	X
SHOW	X	X	X

Syntax



Keywords

The following keywords are valid for the QUERY POOL command:

TYPE()
Required keyword that specifies the type of pool that you want the information to display for. The pool types are mutually exclusive. The command is rejected with a syntax error if multiple types are specified.

FPBP64
Returns information about the Fast Path 64-bit buffer pool, displaying each subpool and subpool extent on a different output line.

ACBIN64
Returns information about the 64-bit cache pools. The 64-bit ACB storage pool is ACBIN64.

SHOW()
Specifies the pool output fields to be returned. There is no default.

This parameter is required when you specify the TYPE(FPBP64) keyword and is optional when you specify the TYPE(ACBIN64) keyword.

ALL
Returns all information for the pool type that you specified.

STATISTICS
Returns specific point-in-time statistics information for the specified pool type.

Usage notes

The QUERY POOL command can be specified only through the OM API.

Output fields

The following table shows the QUERY POOL output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 122. Output fields for the QUERY POOL command

Short label	Long label	Keyword	Meaning
BSIZE	Base Size	N/A	The size (in buffers) of the base section of this subpool, which was initially allocated when the subpool was created.
BUFAVL	Buf_Avl	TYPE(FPBP64)	The number of buffers available for use from this subpool or extents, or the grand total for the entire buffer pool.
BUFUSE	Buf_Use	TYPE(FPBP64)	The number of buffers being used by a process (IMS, dependent region or external thread (such as ODBM)) from this subpool or extents, or the grand total for the entire buffer pool.
CC	CC	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned.
	Csto	TYPE(ACBIN64)	The number of times members were cast out from the 64-bit pool to make room for another. This count is initialized at IMS startup.
DATECR	TimeCreate	TYPE(FPBP64)	The date this subpool or extent was created.
DELETES	Del	TYPE(ACBIN64)	The number of times IMS deleted an ACB member from the 64-bit storage pool. This count is initialized at IMS startup.
ECSA	ECSA_Buf	TYPE(FPBP64)	The ECSA used for the buffers for each subpool and for each extent.
ECSAT	ECSA_Tot	TYPE(FPBP64)	The total amount of ECSA used for each subpool and extents, and the overall ECSA total.
ECSAOT	ECSA_Oth	TYPE(FPBP64)	The amount of ECSA allocated for control blocks.
EXTPER	%Ext	TYPE(FPBP64)	Percentage of the base section to be utilized before an extent is taken. This value might change over time based on buffer usage.
FREE	Free	TYPE(ACBIN64)	The percentage of the pool that has not been reserved for buffers or control data.
FINDS	Gets	TYPE(ACBIN64)	The number of times IMS attempted to retrieve a member from the 64-bit storage pool. This count is initialized at IMS startup.

Table 122. Output fields for the QUERY POOL command (continued)

Short label	Long label	Keyword	Meaning
HITS	Hits	TYPE(ACBIN64)	The number of times a GET request from the 64-bit pool was successful. That number is displayed as a percentage.
ADDS	Isrt	TYPE(ACBIN64)	The number of times IMS inserted an ACB member into the 64-bit storage pool. This count is initialized at IMS startup.
LGBFNM	Lmbr	TYPE(ACBIN64)	The name of the largest member in the 64-bit storage pool.
LGBFTP	Ltype	TYPE(ACBIN64)	The resource type of the largest member in the 64-bit storage pool.
LGBFSZ	Lsize	TYPE(ACBIN64)	The size in kilobytes (KB) of the largest member specified.
MBR	MbrName	N/A	The IMS identifier.
MBRS	Mbrs	TYPE(ACBIN64)	The total number of members in the 64-bit storage pool.
MISSES	Miss	TYPE(ACBIN64)	The number of times a GET request from the 64-bit pool was unsuccessful. That number is displayed as a percentage.
OVERFLOW	Overflow	TYPE(ACBIN64)	The total number of overflow buffers in use in the pool. Overflow buffers are used to store the members that are too large to store in standard 64-bit cache buffers.
POOL	Subpool Name	N/A	Pool name. The internal name of the subpool where *xxxx* is a numeric value: **DBFCxxxx** A common subpool used for DEDB data. The buffers resides in 64-bit addressable storage. **DBFXxxxx** A system subpool used for all other buffer requests, including IMS internal buffers. The buffers resides in ECSA. **ACBIN64** A 64-bit ACB storage pool.
SIZE	Size	TYPE(FPBP64)	The size of the 64-bit buffer or 64-bit storage pool as specified by the user.
SMBFNM	Smbr	TYPE(ACBIN64)	The name of the smallest member in the 64-bit storage pool.
SMBFTP	Stype	TYPE(ACBIN64)	The resource type of the smallest member in the 64-bit storage pool.
SMBFSZ	Ssize	TYPE(ACBIN64)	The size in kilobytes (K) of the smallest member specified.
TOTBUF	Tot_Buf	TYPE(FPBP64)	The total number of buffers in this subpool, including the base section and the extents.

Table 122. Output fields for the QUERY POOL command (continued)

Short label	Long label	Keyword	Meaning
TYPE	Type	TYPE(FPBP64)	Pool type. Type can be one of the following: **G** This row contains overall totals for the entire buffer pool. **A** The total values for the subpool and extents, with the name of the subpool in the SUBPOOL column. **B** This is the base section of the subpool, and does not include the extent values. **E** This is an extent for the subpool, and does not include the base section of the subpool.
USE	%Use	TYPE(FPBP64)	Percentage of the buffers that are currently in use by a process for a subpool or extents.
INUSE	Used	TYPE(ACBIN64)	The percentage of the pool that is used.
B64BT	64b_Buf	TYPE(FPBP64)	The 64 bit storage used for the buffers for each subpool and extent.
B64BTOT	64b_Tot	TYPE(FPBP64)	The total amount of 64 bit storage used for each subpool and extents, and the overall 64 bit storage total.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY POOL command. The OM return and reason codes that might be returned as a result of the QUERY POOL command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 123. Return and reason codes for the QUERY POOL command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See Table 124 on page 263 for details.
X'00000004'	X'00001024'	The command found no 64-bit storage pools defined.
X'00000010'	X'00004014'	The command failed because it is not valid on the RSR tracker.
X'00000010'	X'0000401C'	The command failed because it is not valid on an FDBR region.
X'00000010'	X'00004320'	The user specified not to use the Fast Path 64-bit buffer manager, so there is no output to display.
X'00000010'	X'00004324'	The user disabled Fast Path for this IMS, so there is no data to display.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 124. Completion codes for the QUERY POOL command

Completion code	Completion code text	Meaning
0	X'00000000'	The QUERY POOL command completed successfully.

Completion codes are not issued as part of this QUERY POOL command. The expected output will show available pool or subpool information.

Examples

The following are examples of the QUERY POOL command:

Example 1 for QUERY POOL command

TSO SPOC input:

```
QUERY POOL TYPE(FPBP64) SHOW(ALL)
```

TSO SPOC output:

```
Response for: QRY POOL TYPE(FPBP64) SHOW(ALL)
 Subpool   MbrName   CC   Size Type Tot_Buf Buf_Use Buf_Avl %Use %Ext
 ---------------------------------------------------------------------
 DBF_MAXB SYS3

 DBF_TOTB SYS3                  G       219       0     219

 DBFC0001 SYS3             512  A        40              40

 DBFC0001 SYS3        0         B        40       0      40    0   80
 DBFC0002 SYS3            1024  A        40              40

 DBFC0002 SYS3        0         B        40       0      40    0   40
 DBFC0003 SYS3            2048  A        16              16

 DBFC0003 SYS3        0         B        16       0      16    0  100
 DBFC0004 SYS3            4096  A        47              47

 DBFC0004 SYS3        0         B        47       0      47    0   34
 DBFS0001 SYS3             512  A        32              32

 DBFS0001 SYS3        0         B        32       0      32    0   50
 DBFS0002 SYS3            1024  A        28              28

 DBFS0002 SYS3        0         B        28       0      28    0   28
 DBFS0003 SYS3            2048  A         8               8

 DBFS0003 SYS3        0         B         8       0       8    0  100
 DBFS0004 SYS3            4096  A         8               8

 DBFS0004 SYS3        0         B         8       0       8    0  100
(scrolled right to screen 2)
ECSA_Tot ECSA_Buf ECSA_Oth 64b_Tot 64b_Buf  TimeCreate
-----------------------------------------------------------
                               100M

    62680      92K       4K    280K    280K

        0                     20K

                         17K            20K  2010.088 14:26:14.83
        0                     40K
```

```
                          17K             40K  2010.088 14:26:14.83
        0                         32K

                           7K             32K  2010.088 14:26:14.83
        0                       188K

                          20K            188K  2010.088 14:26:14.83
      16K

                 16K      14K                  2010.088 14:26:14.83
      28K

                 28K      12K                  2010.088 14:26:14.83
      16K

                 16K       3K                  2010.088 14:26:14.83
      32K

                 32K       3K                  2010.088 14:26:14.83
```

OM API input:

```
CMD (QUERY POOL TYPE(FPBP64) SHOW(ALL))
```

OM API output:

```
<cmdrsphdr>
<hdr slbl="POOL" llbl="Subpool" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="right" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
 len="4" dtype="CHAR" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="SIZE" llbl="Size" scope="LCL" sort="n" key="0" scroll="no"
 len="6" dtype="CHAR" align="right" />
<hdr slbl="TYPE" llbl="Type" scope="LCL" sort="a" key="3" scroll="no"
 len="1" dtype="CHAR" align="center" />
<hdr slbl="TOTBUF" llbl="Tot_Buf" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="USE" llbl="Buf_Use" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="ABUF" llbl="Buf_Avl" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="PUSE" llbl="%Use" scope="LCL" sort="n" key="0" scroll="yes"
 len="3" dtype="CHAR" align="right" />
<hdr slbl="EXTPER" llbl="%Ext" scope="LCL" sort="n" key="yes"
 scroll="yes" len="3" dtype="CHAR" align="right" />
<hdr slbl="ECSAT" llbl="ECSA_Tot" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="ECSA" llbl="ECSA_Buf" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="ECSAO" llbl="ECSA_Oth" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="B64BT" llbl="64b_Tot" scope="LCL" sort="n" key="0"
 scroll="n" len="5" dtype="CHAR" align="right" />
<hdr slbl="B64B" llbl="64b_Buf" scope="LCL" sort="n" key="0" scroll="n"
 len="5" dtype="CHAR" align="right" />
<hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left"  skipb="yes"  />
</cmdrsphdr>
<cmdrspdata>
<rsp>POOL(DBF_MAXB)  MBR(SYS3    )           B64BT(  100M)</rsp>
<rsp>POOL(DBF_TOTB)  MBR(SYS3    )           TYPE(G) TOTBUF(  219) USE(
    0) ABUF(  219) ECSAT(62680) ECSA(   92K) ECSAO(    4K) B64B(  280K)
 B64BT(  280K)</rsp>
```

```
<rsp>POOL(DBFS0004)  MBR(SYS3    )           SIZE( 4096) TYPE(A)
 TOTBUF(    8) USE(     ) ABUF(    8) ECSAT(   32K)</rsp>
<rsp>POOL(DBFS0004)  MBR(SYS3    ) CC(   0)  TYPE(B) TOTBUF(    8) USE(
    0) ABUF(    8) PUSE(  0) EXTPER(100) ECSA(   32K) ECSAO(    3K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFC0004)  MBR(SYS3    )           SIZE( 4096) TYPE(A)
 TOTBUF(   47) USE(     ) ABUF(   47) ECSAT(    0) B64BT(  188K)</rsp>
<rsp>POOL(DBFC0004)  MBR(SYS3    ) CC(   0)  TYPE(B) TOTBUF(   47) USE(
    0) ABUF(   47) PUSE(  0) EXTPER( 34) B64B(  188K) ECSAO(   20K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFS0003)  MBR(SYS3    )           SIZE( 2048) TYPE(A)
 TOTBUF(    8) USE(     ) ABUF(    8) ECSAT(   16K)</rsp>
<rsp>POOL(DBFS0003)  MBR(SYS3    ) CC(   0)  TYPE(B) TOTBUF(    8) USE(
    0) ABUF(    8) PUSE(  0) EXTPER(100) ECSA(   16K) ECSAO(    3K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFC0003)  MBR(SYS3    )           SIZE( 2048) TYPE(A)
 TOTBUF(   16) USE(     ) ABUF(   16) ECSAT(    0) B64BT(   32K)</rsp>
<rsp>POOL(DBFC0003)  MBR(SYS3    ) CC(   0)  TYPE(B) TOTBUF(   16) USE(
    0) ABUF(   16) PUSE(  0) EXTPER(100) B64B(   32K) ECSAO(    7K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFS0002)  MBR(SYS3    )           SIZE( 1024) TYPE(A)
 TOTBUF(   28) USE(     ) ABUF(   28) ECSAT(   28K)</rsp>
<rsp>POOL(DBFS0002)  MBR(SYS3    ) CC(   0)  TYPE(B) TOTBUF(   28) USE(
    0) ABUF(   28) PUSE(  0) EXTPER( 28) ECSA(   28K) ECSAO(   12K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFC0002)  MBR(SYS3    )           SIZE( 1024) TYPE(A)
 TOTBUF(   40) USE(     ) ABUF(   40) ECSAT(    0) B64BT(   40K)</rsp>
<rsp>POOL(DBFC0002)  MBR(SYS3    ) CC(   0)  TYPE(B) TOTBUF(   40) USE(
    0) ABUF(   40) PUSE(  0) EXTPER( 40) B64B(   40K) ECSAO(   17K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFS0001)  MBR(SYS3    )           SIZE(  512) TYPE(A)
 TOTBUF(   32) USE(     ) ABUF(   32) ECSAT(   16K)</rsp>
<rsp>POOL(DBFS0001)  MBR(SYS3    ) CC(   0)  TYPE(B) TOTBUF(   32) USE(
    0) ABUF(   32) PUSE(  0) EXTPER( 50) ECSA(   16K) ECSAO(   14K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFC0001)  MBR(SYS3    )           SIZE(  512) TYPE(A)
 TOTBUF(   40) USE(     ) ABUF(   40) ECSAT(    0) B64BT(   20K)</rsp>
<rsp>POOL(DBFC0001)  MBR(SYS3    ) CC(   0)  TYPE(B) TOTBUF(   40) USE(
    0) ABUF(   40) PUSE(  0) EXTPER( 80) B64B(   20K) ECSAO(   17K)
 TMCR(2010.088 14:26:14.83)</rsp>
</cmdrspdata>
```

Explanation: In this example, information about the Fast Path 64-bit DEDB buffer pool is displayed. Each subpool is shown on a different output line.

Example 2 for QUERY POOL command

TSO SPOC input:

```
QRY POOL TYPE(ACBIN64)
```

TSO SPOC output:

```
PoolName Type     MbrName    CC
ACBIN64  Cache64  IMS1        0
```

OM API input:

```
CMD (QRY POOL TYPE(ACBIN64))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
```

```
<statime>2010.090 17:04:30.525184</statime>
<stotime>2010.090 17:04:30.525588</stotime>
<staseq>C5C2C060F8700392</staseq>
<stoseq>C5C2C060F8894DD2</stoseq>
<rqsttkn1>USRT011 10100430</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>POOL            </kwd>
<input>QRY POOL TYPE(ACBIN64) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="POOLNM" llbl="PoolName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TYPE" llbl="Type" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBRNAME" llbl="MbrName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="right" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>POOLNM(ACBIN64 ) TYPE(Cache64 ) MBRNAME(SYS3    ) CC(   0)
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: In this example, information about the 64-bit cache pools is displayed.

Example 3 for QUERY POOL command

TSO SPOC input:

```
QRY POOL TYPE(ACBIN64) SHOW(ALL)
```

TSO SPOC output:

```
PoolName Type     CC Size  Mbrs  Used  Free  Overflow   Gets   Hit  Miss
ACBIN64  Cache64   0   2   3700    25    75         5  10000  9603   397

  Isrt    Del  Lmbr     Ltype  Lsize   Smbr   Stype  Ssize
   300     20  PAYROLL  PSB     2000   DEBIT  INT      100
```

OM API input:

```
CMD (QRY POOL TYPE(ACBIN64) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.090 18:17:16.097203</statime>
<stotime>2010.090 18:17:16.097784</stotime>
<staseq>C5C2D0A44DEB3E15</staseq>
<stoseq>C5C2D0A44E0F8CD5</stoseq>
<rqsttkn1>USRT011 10111716</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
```

```
<cmd>
<master>SYS3    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>POOL            </kwd>
<input>QRY POOL TYPE(ACBIN64) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="POOLNM" llbl="PoolName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TYPE" llbl="Type" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBRNAME" llbl="MbrName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="POOLSZ" llbl="Size" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="MBRS" llbl="Mbrs" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="INUSE" llbl="Used" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="FREE" llbl="Free" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="OVERFLOW" llbl="Overflow" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="FINDS" llbl="Gets" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="HITS" llbl="Hit" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="MISSES" llbl="Miss" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="ADDS" llbl="Isrt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="DELETES" llbl="Del" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="LGBFNM" llbl="Lmbr" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LGBFTP" llbl="Ltype" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LGBFSZ" llbl="Lsize" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="SMBFNM" llbl="Smbr" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SMBFTP" llbl="Stype" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SMBFSZ" llbl="Ssize" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>POOLNM(ACBIN64 ) TYPE(Cache64 ) MBRNAME(SYS3    ) CC(   0)
 POOLSZ(    5120) MBRS(      20) INUSE(       0) FREE(     100)
 OVERFLOW(       0) FINDS(       0) HITS(       0) MISSES(       0)
 ADDS(      14) DELETES(       0) LGBFNM(BMP255  ) LGBFTP(JCB     )
 LGBFSZ(   45056) SMBFNM(DX41SK01) SMBFTP(DMB     ) SMBFSZ(     512)
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: In this example, detailed information about the 64-bit cache pools is displayed.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY POOL command (Commands)

Command keywords and their synonyms (Commands)

QUERY RTC command

Use the QUERY RTC command to query Fast Path routing codes. A routing code can be used by the Fast Path Input Edit/Routing Exit Routine (DBFHAGU0) to route a transaction to a different application program within the same load balancing group.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 269
- "Usage notes" on page 271
- "Output fields" on page 271
- "Return, reason, and completion codes" on page 273
- "Examples" on page 274

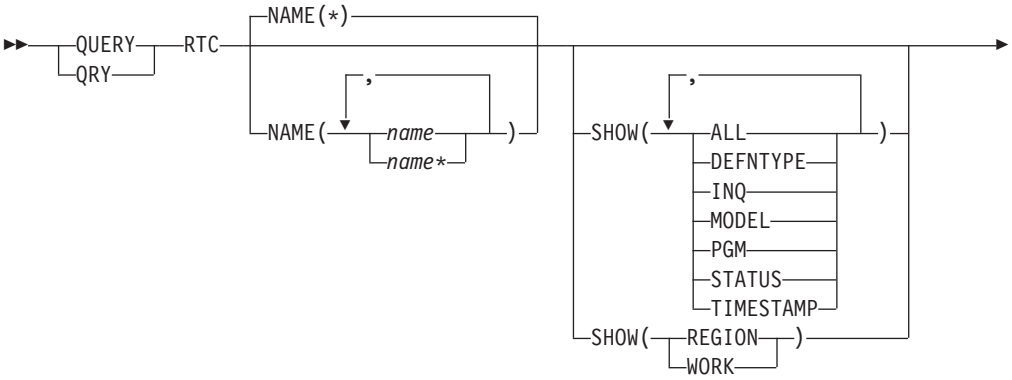
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 125. Valid environments for the QUERY RTC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY RTC	X		X
NAME	X		X
SHOW	X		X
STATUS	X		X

Syntax



Keywords

The following keywords are valid for the QUERY RTC command:

NAME
Specifies the 1-8 character name of the routing code. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all routing code resources.

SHOW
Specifies the routing code output fields to be returned. The routing code is always returned, along with the name of the IMS that created the output and the completion code. The filters that are supported with the SHOW keyword are:

ALL
Returns all information about the routing code itself. Other SHOW keywords can be specified to return information about resources that are related to the routing code.

DEFNTYPE
Definition type that the resource was defined with.

INQ
Inquiry option for transaction messages associated with this routing code.

MODEL
The model name and model type used to create this resource. If the resource is created with all of the attributes defined and no model specified, the model name and model type are blank. The CREATE command specified without the LIKE keyword creates a resource using the default descriptor as a model. The default descriptor is either the IMS descriptor DBFDSRT1 or user-defined. The CREATE command specified with the LIKE keyword creates a resource using a model. The resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

PGM
Program associated with this routing code.

REGION
Regions where the routing code is active.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(REGION) individually.

STATUS
Local routing code status. For a description of the possible routing code status returned, see the STATUS keyword in the "Output fields for the QUERY RTC command" under “Output fields” on page 271.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMECREATE) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

WORK

Work in progress for the routing code specified on NAME parameter and its associated resources. The QRY RTC SHOW(WORK) command can be issued before a DELETE, IMPORT or UPDATE command to check for any work in progress for the specified routing code and any of its associated resources. Any work in progress might cause the subsequent DELETE, IMPORT or UPDATE commands to fail. The QRY RTC SHOW(WORK) command returns the resource name, resource type, and work status for the work in progress for the routing code specified on the NAME parameter or work in progress for an associated resource. If no work is in progress for the specified resource as response line is returned with a work status of blanks.

SHOW(WORK) specified with NAME(*) might take a long time.

Notes:

1. You cannot specify this filter with other SHOW filters; you must specify SHOW(WORK) individually.
2. The QRY RTC SHOW(WORK) command is not valid on an XRF alternate.

STATUS()
Selects routing codes for display that match the NAME parameter and possess at least one of the specified routing code status values.

ACTIVE
Sets the STATUS() filter to return information about active routing codes.

NOTINIT
Sets the STATUS() filter to return information about routing codes that are not initialized and cannot be used.

NOTSCHD
Sets the STATUS() filter to return information about routing codes that are not scheduled.

STOQ
Sets the STATUS() filter to return information about routing codes that are stopped. Transactions that are associated with this routing code are not processed.

Usage notes

This command can be issued only through the Operations Manager API. Fast Path must be installed on the system. This command applies to DB/DC and DCCTL systems. This command is allowed on XRF alternate and RSR tracker systems.

Output fields

The following table shows the QUERY RTC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 126. Output fields for the QUERY RTC command

Short label	Keyword	Scope	Meaning
CC	N/A	LCL	Completion code.
CCTXT	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DFNT	DEFNTYPE	LCL	Definition type, which can be one of the following: **CREATE** Defined by a CREATE command. **IMPORT** Defined by an IMPORT command. **IMS** Defined by IMS. RTCDUMMY is an IMS-defined routing code created by the system definition process if no routing codes are defined. **MODBLKS** Defined by system definition in the MODBLKS data set. **UPDATE** Defined by system definition in the MODBLKS data set, but changed into a dynamic resource by an UPDATE command.
INQ	INQ	LCL	Inquiry transaction associated with routing code (Y) or not (N). **N** The inquiry option is not enabled. **Y** The inquiry option is enabled. The transaction messages associated with this routing code do not cause a change to a database. Programs are prohibited from issuing Insert, Delete, or Replace calls to a database.

Table 126. Output fields for the QUERY RTC command (continued)

Short label	Keyword	Scope	Meaning
LSTT	STATUS	LCL	Local routing code status. **ACTIVE** Routing code is active. **NOTINIT-***xx-reason* Routing code is not initialized and cannot be used. The NOTINIT status is displayed in the format NOTINIT-*xx-reason*, to identify where an error was detected and the reason the routing code was not initialized. *xx* is the reason code that identifies the unique location in one module where this reason code is set. DBFRCTE MACRO defines each reason code that may be set in the routing code bad reason code (field RCTEBADR) and identifies the module that sets it. NOTINIT-00 indicates that the reason is unknown. Call IBM Software Support (Action: 1). *reason* explains the reason code *xx* in abbreviated text format up to 13 characters. For possible reason values and their descriptions, see Table 127 on page 273. **NOTSCHD** Routing code is not scheduled. **STOQ** Routing code is stopped.
MBR	N/A	LCL	IMSplex member that built the output line.
MDLN	MODEL	LCL	Model name. Name of the resource used as a model to create this resource. DBFDSRT1 is the IMS descriptor name for routing codes.
MDLT	MODEL	LCL	Model type, either RSC or DESC. RSC means the resource was created using another resource as a model. DESC means the resource was created using a descriptor as a model.
PGM	PGM	LCL	Program name.
REG	REGION	LCL	Region where routing code is active.
RTC	RTC	LCL	Routing code name.
TMAC	TIMESTAMP	LCL	The time that the resource was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. For a routing code resource, the following actions update the last access time: • When a message is queued to the balancing group using the specified routing code. • CREATE command references the resource as a model.
TMCR	TIMESTAMP	LCL	The time that the routing code was created. This is the result of a CREATE RTC command, IMPORT command that creates the routing code, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMIM	TIMESTAMP	LCL	The time that the resource was last imported, if applicable. The import time is retained across warm start and emergency restart.

Table 126. Output fields for the QUERY RTC command (continued)

Short label	Keyword	Scope	Meaning
TMUP	TIMESTAMP	LCL	The time that the routing code was updated as a result of an UPDATE RTC command that changes a runtime resource definition attribute. This includes attributes set by the SET keyword. The update time is retained across warm start, emergency restart, EXPORT, and IMPORT.
WRK	WORK	LCL	Work is in progress for the routing code or one of its associated resources. The work in progress can be one of the following: **ACTIVE** Routing code is active. **ANOTHER CMD IN PROGRESS** Another command (such as DELETE or UPDATE) to delete or update the routing code is already in progress.

Table 127. Reason information for NOTINIT-xx-reason status

Reason	Meaning
NOPGM	The program that this routing code references is not defined. No program PDIR control block exists. Action: 2.

Note: Actions that can be taken to initialize the routing code are:
1. Call IBM Software Support.
2. Issue the CREATE PGM command to create the program. Issue the UPDATE RTC START(Q) command to complete routing code initialization.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 128. Return and reason codes for the QUERY RTC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000004'	X'00001010'	No resources were found to be returned. The resource name(s) specified may be invalid, or there were no resources that match the filter specified, or there were no resources that had work to display for the SHOW(WORK) specified.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined.

Table 128. Return and reason codes for the QUERY RTC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS OLC, or MODBLKS not defined).
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 129. Completion codes for the QUERY RTC command

Completion code	Completion code text	Meaning
0		Command completed successfully for routing code.
10	NO RESOURCES FOUND	Routing code name is invalid, or the wildcard parameter specified does not match any resource names.

Examples

The following are examples of the QUERY RTC command:

Example 1 for QUERY RTC command

TSO SPOC input:

```
QUERY RTC NAME(%%%%,TXCDJN11,TXBANKI2,RTC0*) SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
Rtcode   MbrName    CC PgmName  Inq LclStat ModelName ModelType TimeCreate>
GFP1     IMS1        0 DCGPSBFP N   NOTSCHD                     2006.286 09:14:5
GFP2     IMS1        0 DCGPSBF2 N   NOTSCHD                     2006.286 09:14:5
RTC00001 IMS1        0 DCGPSBF2 N   NOTSCHD GFP2      RSC       2006.286 13:20:1
RTC00002 IMS1        0 DCGPSBF2 N   NOTSCHD DBFDSRT1  DESC      2006.286 13:20:4
TXBANKI2 IMS1        0 BANKIFP  N   NOTSCHD                     2006.286 09:14:5
TXCDJN11 IMS1        0 DDLTJN11 Y   NOTSCHD                     2006.286 09:14:5
(scrolled to the right screen 2)
Rtcode   MbrName    CC reate TimeUpdate           TimeAccess           TimeImpor
GFP1     IMS1        0 53.98
GFP2     IMS1        0 53.98                     2006.286 13:20:16.89
RTC00001 IMS1        0 16.89
RTC00002 IMS1        0 43.35 2006.286 13:22:25.80
TXBANKI2 IMS1        0 53.98
TXCDJN11 IMS1        0 53.98
(scrolled to the right screen 3)
Rtcode   MbrName    CC TimeAccess           TimeImport           DefnType
GFP1     IMS1        0                                        MODBLKS
GFP2     IMS1        0 2006.286 13:20:16.89                   MODBLKS
RTC00001 IMS1        0                                        CREATE
RTC00002 IMS1        0                                        CREATE
TXBANKI2 IMS1        0                                        MODBLKS
TXCDJN11 IMS1        0                                        MODBLKS
```

OM API input:

```
CMD(QUERY RTC NAME(%%%%,TXCDJN11,TXBANKI2,RTC0*) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.286 20:23:24.453527</statime>
<stotime>2006.286 20:23:24.453976</stotime>
<staseq>BF8C7581F7297788</staseq>
<stoseq>BF8C7581F74585C8</stoseq>
<rqsttkn1>USRT011 10132324</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>RTC             </kwd>
<input>QRY RTC NAME(%%%%,TXCDJN11,TXBANKI2,RTC0*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="RTC" llbl="Rtcode" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
 len="4" dtype="CHAR" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="PGM" llbl="PgmName" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="INQ" llbl="Inq" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" />
<hdr slbl="MDLN" llbl="ModelName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MDLT" llbl="ModelType" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMUP" llbl="TimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMAC" llbl="TimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="TimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="DFNT" llbl="DefnType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RTC(GFP1    )  MBR(IMS1    ) CC(   0)  PGM(DCGPSBFP) INQ(N)
 LSTT(NOTSCHD) TMCR(2006.286 09:14:53.98) TMUP(                    )
TMAC(                    ) TMIM(                    ) DFNT(MODBLKS )
 </rsp>
 <rsp>RTC(GFP2    )  MBR(IMS1    ) CC(   0)  PGM(DCGPSBF2) INQ(N)
  LSTT(NOTSCHD) TMCR(2006.286 09:14:53.98) TMUP(                    )
  TMAC(2006.286 13:20:16.89) TMIM(                    ) DFNT(MODBLKS )
 </rsp>
 <rsp>RTC(TXCDJN11)  MBR(IMS1    ) CC(   0)  PGM(DDLTJN11) INQ(Y)
  LSTT(NOTSCHD) TMCR(2006.286 09:14:53.98) TMUP(                    )
  TMAC(                    ) TMIM(                    ) DFNT(MODBLKS )
 </rsp>
```

```
<rsp>RTC(TXBANKI2)  MBR(IMS1    ) CC(   0)  PGM(BANKIFP ) INQ(N)
 LSTT(NOTSCHD) TMCR(2006.286 09:14:53.98) TMUP(                    )
 TMAC(                    ) TMIM(                    ) DFNT(MODBLKS )
</rsp>
<rsp>RTC(RTC00001)  MBR(IMS1    ) CC(   0)  PGM(DCGPSBF2) INQ(N)
 LSTT(NOTSCHD) MDLN(GFP2    ) MDLT(RSC) TMCR(2006.286 13:20:16.89)
 TMUP(                    ) TMAC(                    ) TMIM(
        ) DFNT(CREATE  )</rsp>
<rsp>RTC(RTC00002)  MBR(IMS1    ) CC(   0)  PGM(DCGPSBF2) INQ(N)
 LSTT(NOTSCHD) MDLN(DBFDSRT1) MDLT(DESC) TMCR(2006.286 13:20:43.35)
 TMUP(2006.286 13:22:25.80) TMAC(                    ) TMIM(
        ) DFNT(CREATE  )</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY RTC command is specified with SHOW(ALL), so all output fields are returned for the specified routing codes. All of the routing code output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The routing code name, the member name that built the line of output, and the completion code are displayed on every screen. Routing code wildcard parameter %%%% causes all routing codes with 4-character names to be displayed. The routing codes that were generated in stage 1 MODBLKS generation are created when IMS cold starts and defined with a definition type of MODBLKS. Routing codes RTC00001 and RTC00002 were created dynamically using CREATE RTC commands and display a definition type of CREATE. RTC00001 was created using routing code GFP2 as the model and RTC00002 was created using default descriptor DBFDSRT1. Routing code GFP2 displays the last access time stamp, from when it was referred to by the CREATE RTC command that created routing code RTC00001, using routing code GFP2 as the model. Routing code RTC00002 shows the last update time, because its program was changed by an UPDATE RTC command since it was created.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY MODIFY command (Commands)

/DISPLAY RTCODE command (Commands)

/DISPLAY STATUS command (Commands)

Command keywords and their synonyms (Commands)

QUERY RTCDESC command

Use the QUERY RTCDESC command to query Fast Path routing descriptors. A descriptor is a model that can be used to create descriptors.

Subsections:

- "Environment" on page 277
- "Syntax" on page 277
- "Keywords" on page 277
- "Usage notes" on page 278
- "Output fields" on page 279
- "Return, reason, and completion codes" on page 280
- "Example" on page 281

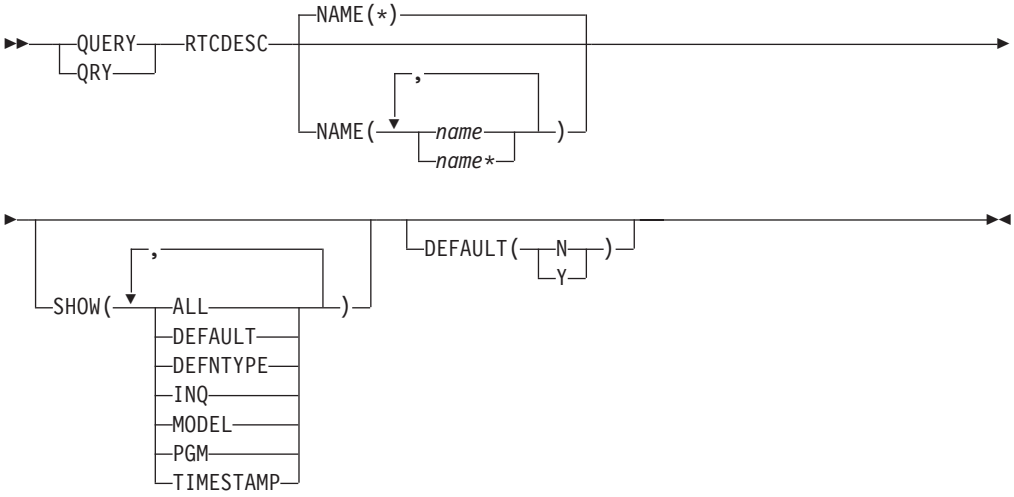
Environment

The following table lists the environments (DB/BC, DBCTL and DCCTL) in which you can use the commands and keywords.

Table 130. Valid environments for the QUERY RTCDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY RTCDESC	X		X
NAME	X		X
SHOW	X		X
DEFAULT	X		X

Syntax



Keywords

The following keywords are valid for the QUERY RTCDESC command:

NAME
Specifies the 1-8 character name of the routing code descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all routing code descriptor resources.

DEFAULT
Selects routing code descriptors for display that possess the default value specified. DEFAULT(Y) displays the one and only default descriptor. DEFAULT(N) displays all the other descriptors that are not the default.

SHOW
Specifies the routing code descriptor output fields to be returned. The routing code descriptor name is always returned, along with the name of the IMS that created the output and the completion code. The filters supported with the SHOW keyword are:

ALL
Returns all information about the routing code descriptor itself. Other SHOW keywords can be specified to return information about resources related to the routing code descriptor.

DEFAULT
Default descriptor option.

DEFNTYPE
Definition type that the descriptor was defined with.

INQ
Inquiry option for transaction messages that are associated with this routing code descriptor.

MODEL
The model name and model type used to create this descriptor. If the descriptor is created with all of the attributes defined and no model specified, the model name and model type are blank. The CREATE command specified without the LIKE keyword creates a descriptor using the default descriptor as a model. The default descriptor is either the IMS descriptor DBFDSRT1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor by using a model. The descriptor is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

PGM
Program that is associated with this routing code.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMECREATE) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

Usage notes

This command can be issued only through the Operations Manager API. Fast Path must be installed on the system. This command applies to DB/DC and DCCTL systems. This command is allowed on XRF alternate and RSR tracker systems. The QUERY RTCDESC command is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

Output fields

The following table shows the QUERY RTCDESC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 131. Output fields for the QUERY RTCDESC command

Short label	Keyword	Scope	Meaning
CC	N/A	LCL	Completion code.
CCTXT	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DESC	RTCDESC	LCL	Routing code descriptor name.
DFNT	DEFNTYPE	LCL	Definition type, which can be one of the following: **CREATE** Defined by a CREATE command. **IMPORT** Defined by an IMPORT command. **IMS** Defined by IMS. DBFDSRT1 is an IMS-defined routing code descriptor containing the default routing code values.
DFLT	DEFAULT	LCL	Default descriptor (Y) or not (N). **N** The descriptor is not the default. **Y** The descriptor is the default. When a descriptor is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a default routing code descriptor called DBFDSRT1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor.
INQ	INQ	LCL	Inquiry transaction associated with routing code (Y) or not (N). **N** The inquiry option is not enabled. **Y** The inquiry option is enabled. The transaction messages associated with this routing code do not cause a change to a database. Programs are prohibited from issuing Insert, Delete, or Replace calls to a database.
MBR	N/A	LCL	IMSplex member that built the output line.

Table 131. Output fields for the QUERY RTCDESC command (continued)

Short label	Keyword	Scope	Meaning
MDLN	MODEL	LCL	Model name. Name of the descriptor used as a model to create this resource. DBFDSRT1 is the IMS descriptor name for routing codes.
MDLT	MODEL	LCL	Model type, either RSC or DESC. RSC means that the resource was created using another resource as a model. DESC means that the resource was created using a descriptor as a model.
PGM	PGM	LCL	Program name.
TMAC	TIMESTAMP	LCL	The time that the routing code descriptor was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. For a routing code descriptor, when the CREATE command references the descriptor as a model, the last access time is updated.
TMCR	TIMESTAMP	LCL	The time that the routing code descriptor was created. This is the result of a CREATE RTCDESC command, IMPORT command that creates the routing code, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMIM	TIMESTAMP	LCL	The time that the routing code descriptor was last imported, if applicable. The import time is retained across warm start and emergency restart.
TMUP	TIMESTAMP	LCL	The time that the routing code descriptor was updated as a result of an UPDATE RTCDESC command that changes a definitional attribute. This includes attributes set by the SET keyword. The update time is retained across warm start, emergency restart, EXPORT, and IMPORT.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 132. Return and reason codes for the QUERY RTCDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000004'	X'00001010'	No resources were found to be returned. The resource names specified might be invalid, or there were no resources that match the filter specified, or there were no resources that had work to display for the SHOW(WORK) specified.

Table 132. Return and reason codes for the QUERY RTCDESC command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS OLC, or MODBLKS not defined).
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 133. Completion codes for the QUERY RTCDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for routing code or routing code descriptor.
10	NO RESOURCES FOUND	Routing code or routing code descriptor name is invalid, or the wildcard parameter specified does not match any resource names.

Example

The following is an example of the QUERY RTCDESC command:

Example 1 for QUERY RTCDESC command

TSO SPOC input:

```
QRY RTCDESC SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
DescName MbrName    CC PgmName  Inq ModelName ModelType Dflt TimeCreate>
DBFDSRT1 IMS1        0          N                       Y    2006.286 09:14:53.9

(scrolled to the right screen 2)
DescName MbrName    CC te TimeUpdate           TimeAccess           TimeImport>
DBFDSRT1 IMS1        0 98                      2006.286 13:20:43.35

(scrolled to the right screen 3)
DescName MbrName    CC TimeAccess           TimeImport           DefnType
DBFDSRT1 IMS1        0 2006.286 13:20:43.35                      IMS
```

OM API input:

```
CMD(QUERY RTCDESC SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.286 20:41:07.204698</statime>
<stotime>2006.286 20:41:07.205040</stotime>
<staseq>BF8C79777BE5A48E</staseq>
<stoseq>BF8C79777BFB078E</stoseq>
<rqsttkn1>USRT011 10134107</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>RTCDESC         </kwd>
<input>QRY RTCDESC SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
 len="4" dtype="CHAR" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="PGM" llbl="PgmName" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="INQ" llbl="Inq" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
<hdr slbl="MDLN" llbl="ModelName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MDLT" llbl="ModelType" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="DFLT" llbl="Dflt" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
<hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMUP" llbl="TimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMAC" llbl="TimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="TimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="DFNT" llbl="DefnType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DBFDSRT1) MBR(IMS1    ) CC(   0)  PGM(        ) INQ(N)
 DFLT(Y) TMCR(2006.286 09:14:53.98) TMUP(                 )
TMAC(2006.286 13:20:43.35) TMIM(                   ) DFNT(IMS     )
 </rsp>
 </cmdrspdata>
 </imsout>
```

Explanation: All routing code descriptors are returned with all output fields. All of the routing code descriptor output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The routing code descriptor name,

the member name that built the line of output, and the completion code are displayed on every screen. IMS-defined descriptor DBFDSRT1 contains the IMS default routing code values.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY STRUCTURE command

Use the QUERY STRUCTURE command to display information about IMS coupling facility structures used by members of an IMSplex.

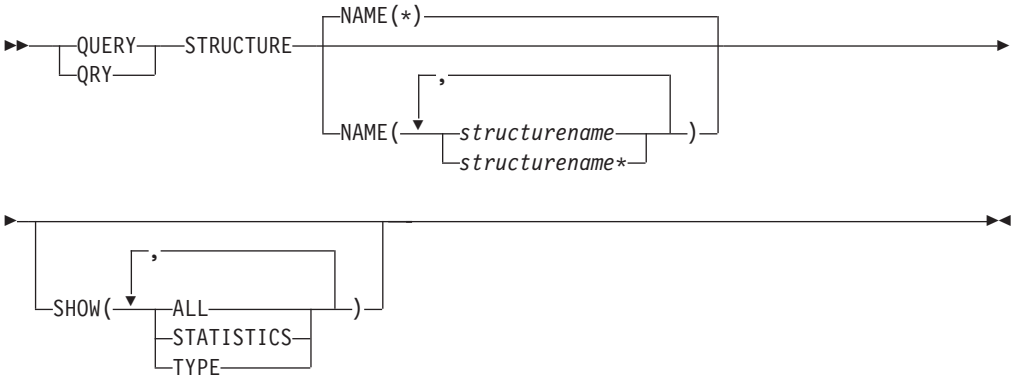
Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 284
- "Output fields" on page 284
- "Return, reason, and completion codes" on page 285
- "Examples" on page 286

Environment

There are no environment indicators for the QUERY STRUCTURE command itself, because it does not run in the address space of any IMS control or dependent region. QUERY STRUCTURE is processed in an RM command processing environment.

Syntax



Keywords

The following keywords are valid for the QUERY STRUCTURE command:

NAME()
Specifies the names of the structures for which information is to be returned.

The structure name can be a generic parameter, to enable easy specification of a group of structures whose names match a generic parameter mask.

SHOW()
Specifies the output fields to be returned. If SHOW is not specified, only the structure names, IMSplex member that builds the output line, and completion codes are returned. This provides a method for a system management application to obtain a list of all structure names. This can be used to determine the resource structure that is managed by RM. The parameters supported with the SHOW keyword are as follows:

ALL
Returns all the output fields.

STATISTICS
Displays statistics information for the structures that match the specification in the NAME() parameter.

TYPE
Returns the type of the specified structure. For example, the type might be RSRC, which identifies a resource structure.

Usage notes

This command is supported only by Resource Manager (RM) to return information about a resource structure.

The command syntax for this command is defined in Extensible Markup Language (XML) and is available to automation programs which communicate with Operations Manager (OM).

Output fields

The following table contains information about the output fields for QUERY STRUCTURE. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 134. Output fields for the QUERY STRUCTURE command

Short label	Long label	Keyword	Meaning
STRNM	StructureName	N/A	Resource structure name. The structure name is always returned.
MBR	MbrName	N/A	IMSplex member that built the output line. The RM identifier of the RM that built the output line.
CC	CC	N/A	Completion code for the line of output. The completion code is always returned.

Table 134. Output fields for the QUERY STRUCTURE command (continued)

Short label	Long label	Keyword	Meaning
TYP	Type	TYPE	Structure type. RSRC indicates the RM resource structure.
LEA	LeAlloc	STATISTICS	Number of list entries that are allocated in the structure.
LEI	LeInuse	STATISTICS	Number of list entries in use in the structure.
ELMA	ElmAlloc	STATISTICS	Number of data elements that are allocated in the structure.
ELMI	ElmInuse	STATISTICS	Number of data elements in use in the structure.
RATIO	LE/EL	STATISTICS	Entry to element ratio. It is in the format of list entries/data elements.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY STRUCTURE command are standard for all commands entered through the OM API.

The following table contains the return and reason codes that can be returned to OM from a QUERY STRUCTURE command.

Table 135. Return and reason codes for the QUERY STRUCTURE command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY STRUCTURE command completed successfully.
X'0300000C'	X'00003000'	The QUERY STRUCTURE command is successful for at least one resource name. The QUERY STRUCTURE command is not successful for one or more resource names. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY STRUCTURE command are listed in the QUERY STRUCTURE completion code table.
X'0300000C'	X'00003004'	No requests were successful. The resource names specified might be invalid, or there were no resources that match the filter specified.
X'03000014'	X'0000502C'	The QUERY STRUCTURE command processing terminated. RM was unable to obtain storage for the command output header.
X'03000014'	X'00005030'	The QUERY STRUCTURE command processing terminated. RM was unable to obtain storage for the command output response.
X'03000014'	X'00005200'	The QUERY STRUCTURE command processing terminated because of an unexpected Common Queue Server (CQS) error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains the completion codes that can be returned on a QUERY STRUCTURE command.

Table 136. Completion codes for the QUERY STRUCTURE command

Completion code	Meaning
0	The QUERY STRUCTURE command completed successfully for the resource.
4	The structure name is unknown to the client that is processing the request. The structure name might have been typed in error, or the structure might not be defined or allocated at this time. If this is a wildcard request there were no matches for the name. Confirm the correct spelling of the structure name is specified on the command.

Examples

The following is an example of the QUERY STRUCTURE command:

Example 1 for QUERY STRUCTURE command

This command displays all of the resource structures in the IMSplex and their statistics. Only one resource structure is defined, IMSRSRC01. The number of list entries allocated on the resource structure is 3577, the number of list entries in use on the structure is 676. The number of data elements allocated is 3574, the number of data elements in use is 24. The list entry to data element ratio on the resource structure is one to one. Not many list entries or data elements are in use on the resource structure, so the resource structure is not approaching full.

TSO SPOC input:

```
QRY STRUCTURE SHOW(STATISTICS)
```

TSO SPOC output:

```
StructureName    MbrName    CC LeAlloc LeInuse ElmAlloc ElmInuse LE/EL
IMSRSRC01        RM1RM       0    3577     676     3574       24 0001/0001
```

OM API input:

```
CMD (QRY STRUCTURE SHOW(STATISTICS))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1</xmlvsn>
<statime>2002.16314:31:34.901057</statime>
<stotime>2002.16314:31:34.941134</stotime>
<staseq>B7C49C9433D410C1</staseq>
<stoseq>B7C49C943D9CEC44</stoseq>
<rqsttkn1>USRT011 10073134</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>RM1RM   </master>
<userid>USRT011</userid>
<verb>QRY </verb>
<kwd>STRUCTURE</kwd>
<input>QUERY STRUCTURE SHOW(ALL)</input>
</cmd>
```

```
<cmdrsphdr>
<hdr slbl="STRNM" llbl="StructureName" scope="LCL" sort="A" key="1" scroll="NO"
len="16" dtype="CHAR" align="left"/>
<hdr slbl="MBR"   llbl="MbrName"  scope="LCL" sort="N" key="0" scroll="NO" len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"    llbl="CC"       scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="TYP"   llbl="Type"     scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="LEA"   llbl="LeAlloc"  scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="LEI"   llbl="LeInuse"  scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="ELMA"  llbl="ElmAlloc" scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="ELMI"  llbl="ElmInuse" scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="RATIO" llbl="LE/EL"    scope="LCL" sort="N" key="0" scroll="YES" len="9"
dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>STRNM(IMSRSRC01) MBR(RM1RM) CC(   0) TYP(RSRC) LEA(3577) LEI( 676) ELMA(3574)
ELMI(  24) RATIO(0001/0001)</rsp>
</cmdrspdata>
</imsout>
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY TRAN command

Use the QUERY TRAN command to display information about transactions (for example, class, status, queue count, and others) across the IMSplex. This command can be specified only through the OM API and is valid on an XRF alternate.

Subsections:

- "Environment"
- "Syntax" on page 288
- "Keywords" on page 289
- "Usage notes" on page 298
- "Equivalent IMS type-1 commands" on page 298
- "Output fields" on page 299
- "Return, reason, and completion codes" on page 310
- "Examples" on page 313

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY TRAN command and keywords can be issued.

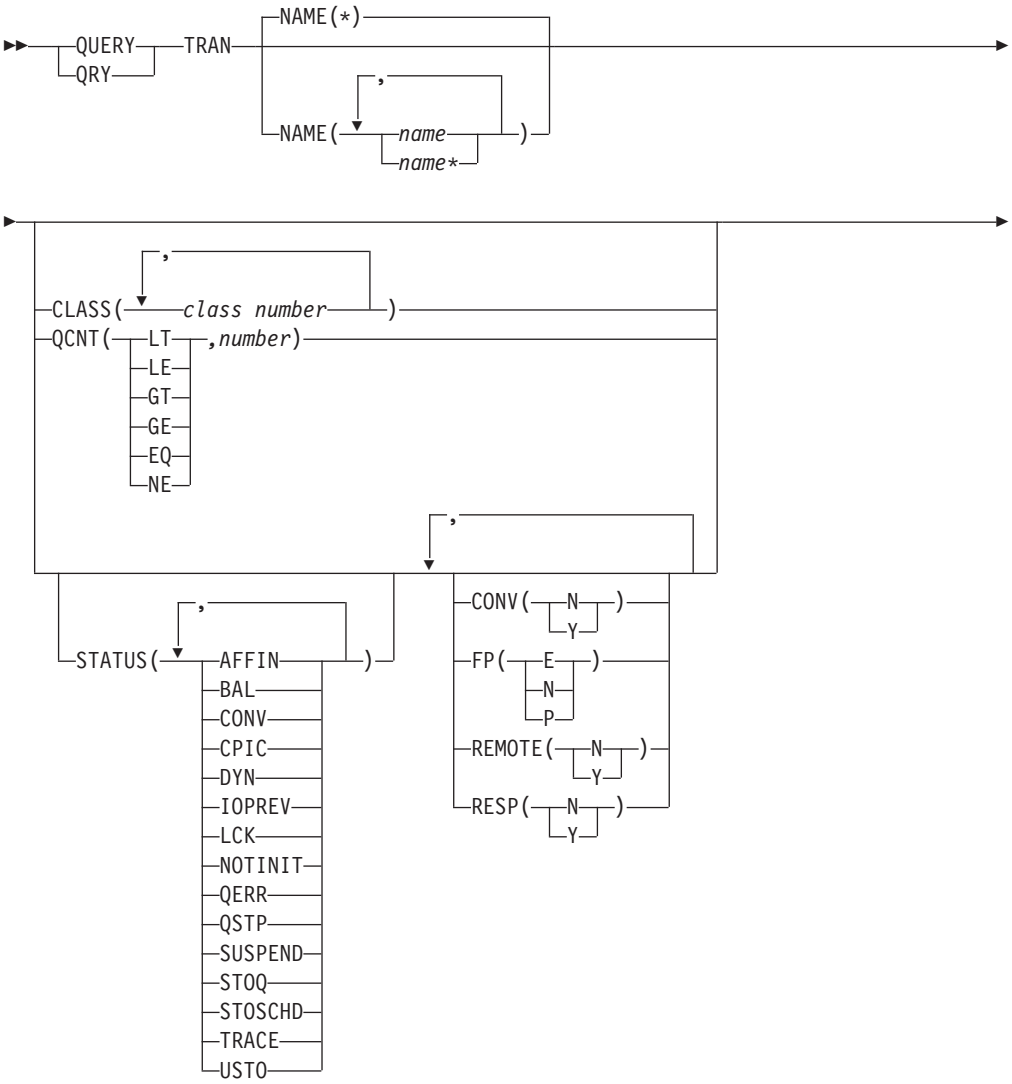
Table 137. Valid environments for the QUERY TRAN command and keywords

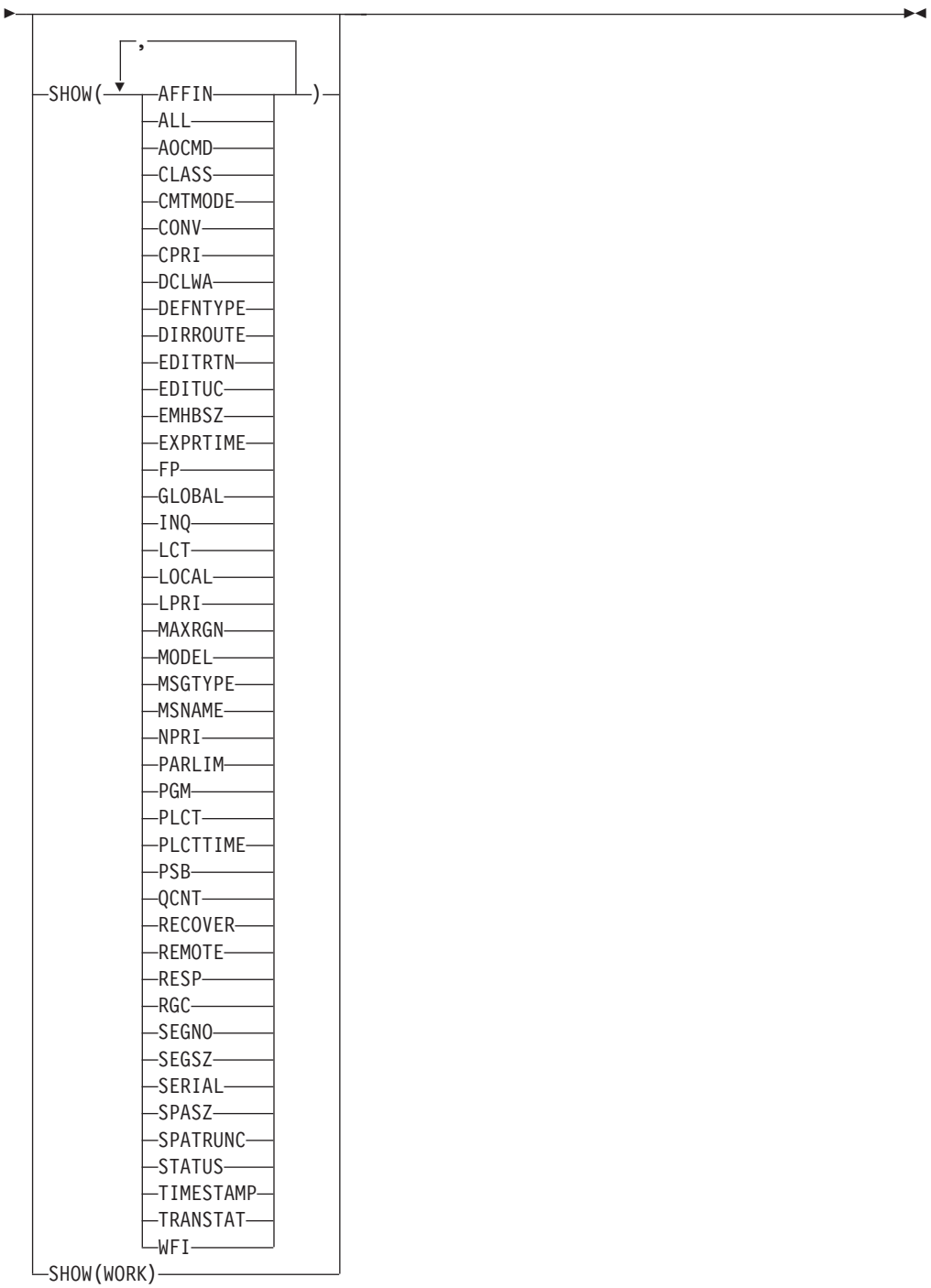
Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY TRAN	X		X

Table 137. Valid environments for the QUERY TRAN command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
CLASS	X		X
CONV	X		X
FP	X		X
NAME	X		X
QCNT	X		X
REMOTE	X		X
RESP	X		X
SHOW	X		X
STATUS	X		X

Syntax





Keywords

The following keywords are valid for the QUERY TRAN command:

CLASS()
Displays transactions that possess at least one of the specified classes. This keyword allows for additional filtering by CLASS value. If no filter is specified (such as STATUS, CLASS or QCNT), all the transactions matching the transaction name are returned.

The output returned when the CLASS filter is specified includes the class value of the transaction that caused the transaction name to be displayed even if the SHOW(CLASS) option is not specified.

CONV()
Selects transactions for display that possess the conversational attributes specified.

FP()
Selects transactions for display that possess the Fast Path option specified. If more than one FP option is specified, selects transactions for display that possess at least one of the Fast Path options specified.

NAME()
Specifies the 1-8 character name of the transaction. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all transactions. If no filter is specified (such as STATUS, CLASS or QCNT), all the transactions matching the transaction name are returned.

QCNT()
Selects transactions that have a queue count less than (LT), less than or equal to (LE), greater than (GT), greater than or equal to (GE), equal to (EQ), or not equal to (NE) the specified numbers. The specified number cannot be a 1 when LT is specified. This keyword allows for additional filtering by QCNT value.

Transactions with a queue count of 0 are not returned when the QCNT filter is specified. When a filter of QCNT(LT,*n*) is specified, transactions with a queue count greater than 0 and less than *n* are returned. If no filter is specified (such as STATUS, CLASS or QCNT), all the transactions matching the transaction name are returned.

The QCNT filter is valid in both a shared-queues environment and a non-shared-queues environment.

In a shared-queues environment, if QCNT is specified, the performance implication is that the shared queues are read. In this environment, the QUERY TRAN QCNT command is processed only by the master IMS as the queues are global. The command master returns all the transactions on the shared queues that match the queue count filter specified. If QCNT is specified with a wildcard transaction name, the performance implication is that all the shared-queues transaction messages on the Coupling Facility must be read.

In a non-shared-queues environment, the local queue count values are used to determine the transactions to be displayed. In this environment, the QUERY TRAN QCNT command is processed by each IMS the command is routed to as the queues are local. Each IMS returns all the transactions it found locally that match the queue count filter specified.

The output returned when the QCNT filter is specified includes the queue count of the transaction that caused the transaction name to be displayed even if the SHOW(QCNT) option is not specified.

REMOTE()
Selects transactions for display that possess the remote option specified.

RESP()
Selects transactions for display that possess the response mode option specified.

SHOW()
Specifies the transaction output fields to be returned. The transaction name is always returned along with the name of the IMS that created the output and

the completion code. If SHOW is not specified, only the transaction names are returned if the QCNT, CLASS, or STATUS filter is not specified. This keyword provides a method for a system management application to obtain a list of transactions that match the transaction names that are currently known in the IMSplex.

Important: The only SHOW option supported when the QCNT() filter is specified is the AFFIN option. No other SHOW options are supported with the QCNT() filter because of performance reasons.

AFFIN
IMS affinity of the messages on the shared queues.

If the AFFIN option is used as a status filter, IMS shows all the transactions that have affinity status.

If the SHOW(ALL) or SHOW(STATUS) keyword is specified, and if a transaction has affinity status, AFFIN is displayed in the LclStat column.

If the SHOW(AFFIN) keyword is used with the QCNT filter, messages on the shared queue are displayed with their AFFIN value.

The affinity that is shown when the SHOW(AFFIN) keyword is specified is valid only with the QCNT filter and is ignored for the other variations of the QUERY TRAN command.

The AFFIN option is valid only in a shared-queues environment and is ignored in a non-shared-queues environment.

ALL
Returns all information about the transaction itself. Other SHOW keywords can be specified to return information about resources that are related to the transaction.

Global values are returned only for those status fields and attributes for which global information is kept for the IMSplex.

AOCMD
Specifies that you want the AOI option returned which indicates whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call.

CLASS
Scheduling class used to determine which message regions can process the transaction locally on a particular IMS.

CMTMODE
Specifies when database updates and non-express output messages are committed. This operand affects emergency restart.

CONV
Conversation option.

CPRI
The current priority. The current priority is the normal priority, when the transaction queue count is less than the limit count. The current priority is raised to the limit priority if the transaction queue count is equal to, or exceeds, the limit cover.

DCLWA
Log write-ahead option.

DEFNTYPE
Definition type, which describes how the resource was defined.

DIRROUTE
MSC directed routing option.

EDITRTN
Input edit routine that edits messages before the program receives the message.

EDITUC
Uppercase translation option of the input data.

EMHBSZ
EMH buffer size that is required to run the Fast Path transaction.

EXPRTIME
Transaction expiration time in seconds.

FP Fast Path option.

GLOBAL
For output fields that have both local and global values, this option returns only the global values. If used with another SHOW keyword to request a specific output field, this OPTION requests that only the global value of the specified output field is returned. IMS retrieves global information from the source that maintains the data. For example, RM maintains some global information, and CQS maintains other global information. Global output is returned only by the command master. The QUERY TRAN command can return global information from the RM resource structure if global status is maintained for transactions.

The QUERY TRAN command returns global information if you specified that the IMSplex maintain global transaction status in RM. You can specify this during IMS initialization in either the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSTRAN(Y)). You can also change it dynamically using the UPD IMS SET(PLEXPARM(GSTSTRAN(Y)) command. If you do not specify that global database status is to be maintained, the GLOBAL keyword is processed, and the global status is not updated.

INQ
Inquiry option.

LCT
Limit count. Specifies the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction. The limit count value can range from 1 through 65 535.

The limit count value is ignored for a transaction that is processed by a BMP.

The limit count value is ignored in a shared-queues environment.

LOCAL
For output fields that have both local and global values, this option returns only the local values. If used with another SHOW keyword to request a specific output field, this option requests that only the local value of the specified output field is returned. Local output is returned by each IMS that processes the command.

LPRI

Limit priority. The scheduling priority to which this transaction is raised when the numbers of input transactions enqueued and waiting to be processes is equal to or greater than the limit count value. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class.

The limit priority value is ignored for a transaction that is processed by a BMP.

MAXRGN

Maximum region count. This limits the number of message processing program (MPP) regions that can be concurrently schedule to process a transaction. When the number of MPP regions is not limited, one transaction might monopolize all available regions. MAXRGN(0) means that no limit is imposed.

MODEL

The model name and model type used to create this resource. The model name and model type are blank for IMS-defined resources and descriptors and queue-only transactions created by the DFSINSX0 exit. The CREATE command specified without the LIKE keyword creates a resource using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSTR1 or user-defined. The CREATE command specified with the LIKE keyword creates a resource using a model. The resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource. The model name is blank if the resource is IMS-defined.

MSGTYPE

Message segment type (single or multiple segment). It specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing.

If MSC-directed routing is used in a multiple IMS system configuration, IMS does not ensure that both the message and the transaction destined to process that message are either single segment or multiple segments.

MSNAME

The logical link path name, remote system ID, and local system ID are returned.

Logical link path name in a multiple IMS system configuration (MSC). A logical link path is a path between any two IMS systems. The IMS systems are identified by the remote system ID and the local system ID associated with the logical link path. The remote system ID identifies the system in which messages using this path are to be processed. The local system ID identifies this system.

The remote system ID (SIDR) identifies the IMS system on which the application program runs. A value of 0 means MSC is not enabled on this system. The local system ID and remote system ID are the same for local transactions.

The local system ID (SIDL) identifies the originating system to which responses are returned. A value of 0 means MSC is not enabled on this system. The local system ID and remote system ID are the same for local transactions.

NPRI
Normal scheduling priority. The scheduling priority is an attribute that is used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The normal priority is assigned to the transaction as the scheduling priority when the number of input transactions that are enqueued and waiting to be processed is less than the limit count value.

The normal priority value is ignored for a transaction that is processed by a BMP.

PARLIM
Parallel processing limit count. This value is the maximum number of messages that can currently be queued, but not yet processed, by each active message region that is currently scheduled for this transaction. This value is the threshold value to be used when the associated program is defined with a scheduling type of parallel. An additional region is scheduled whenever the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions that are currently scheduled for this transaction.

PARLIM(0) indicates that any input message can cause a new region to be scheduled. PARLIM(65535) indicates that parallel processing is disabled and IMS allows the transaction to be scheduled in only one region at a time.

PGM
The name of the program that is associated with this transaction. This name matches the PSB name in ACBLIB. SHOW(PSB) is an alias of SHOW(PGM).

PLCT
Processing limit count. This value is the maximum number of messages that are sent to the application program by the IMS for processing without reloading the application program.

PLCT(0) means that a maximum of one message is sent to the application program at a single program load.

PLCT(65535) means that no limit is placed on the number of messages that are processed at a single program load.

PLCTTIME
Processing limit count time. This value is the amount of time (in hundredths of seconds) allowable to process a single transaction (or message). The number specifies the maximum CPU time allowed for each message to be processed in the message processing region.

For the Fast Path potential transactions that are defined in the system definition TRANSACT macro, the value shown is the CPU-time-per-transaction number, which is specified on the PROCLIM keyword, multiplied by 100.

PLCTTIME(6553500) means that no time limit is placed on the application program.

PSB
The name of the program that is associated with this transaction. This name matches the PSB name in ACBLIB. SHOW(PSB) is an alias of SHOW(PGM).

QCNT
Transaction message queue count.

In a non-shared-queues environment, the local transaction message queue count is returned. This count represents the number of messages on the IMS system where the command is issued.

In a shared-queues environment, both the local transaction message queue count and the global transaction message queue count are returned.

- The local queue counts value that is returned in this environment represents the messages being processed by the IMS system where this command is issued. Do not use local queue counts for queue evaluation purposes.
- The global transaction message queue count represents the current message queue count on the shared queues structure. This count represents the number of messages that can be processed by the IMS system where the command is issued. This count includes messages that can be processed by any IMS system (messages with no affinity), plus messages that can be processed by the IMS system where the command is issued (messages with an affinity to the IMS system that issued the command).

RECOVER
Recovery option.

REMOTE
Remote option.

RESP
Response option.

RGC
Number of regions the transaction is currently scheduled in the local IMS.

SEGNO
Segment number. This value is the maximum number of application program output segments that are allowed into the message queues per Get Unique (GU) call from the application program.

SEGNO(0) means that the number of segments is not checked by the online system at the execution time.

SEGSZ
Segment size. This value is the maximum number of bytes allowed in any one output segment.

SEGSZ(0) means that the segment size is not checked by the online system at execution time.

SERIAL
Serial option.

SPASZ
Scratchpad area size for a conversational transaction.

SPATRUNC

The SPA data truncation option indicates whether the SPA data should be truncated or preserved across a program switch to a transaction that is defined with a smaller SPA.

When a conversation initially starts, and when a program is switched, the SPATRUNC option is checked and set or reset as specified. When the option is set, it remains set for the life of the conversation, or until a program switch occurs to a transaction that specifies the option is to be reset.

When a program switch occurs, the truncated data option for the new transaction is first checked, and that specification is set for the conversation and is used for the SPA inserted into the output message. If the option is not specified for the new transaction, the option currently in effect for the conversation is used.

STATUS
Local transaction status. For a description of the possible transaction status returned, see the STATUS keyword in the "Output fields for the QUERY TRAN command" table under “Output fields” on page 299.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

TRANSTAT
Transaction level statistics option.

WFI
Wait-for-input option.

WORK
Work in progress for the transaction specified on the NAME parameter and its associated resources. The QUERY TRAN SHOW(WORK) command can be issued before a DELETE, IMPORT or UPDATE command to check for any work in progress for the specified transaction and any of its associated resources. Any work in progress might cause the subsequent DELETE, IMPORT or UPDATE commands to fail. If no work is in progress for the specified resource, a response line is returned with a work status of blanks.

Notes:

1. SHOW(WORK) specified with NAME(*) might have a performance impact on the processing of the command.

2. You cannot specify this filter with other SHOW filters; you must specify SHOW(WORK) individually.
3. The QRY TRAN SHOW(WORK) command is not valid on an XRF alternate.

STATUS()
Selects transactions for display that possess at least one of the specified transaction statuses. This keyword allows for additional filtering by transaction status. If a STATUS, CLASS, or QCNT filter is not specified, all the transactions matching the transaction name are returned.

The output returned when the STATUS filter is specified includes the status of the transaction that caused the transaction name to be displayed even if the SHOW(STATUS) option is not specified.

Filtering is based only on local status, even if RM maintains global status.

AFFIN
Transaction is registered with local affinity with IMS.

BAL
Transaction is eligible for load balancing (for example, with parallel limits specified).

CPIC
This CPI-C transaction was built dynamically on this IMS system and can process only on this IMS system.

DYN
Transaction was built in a shared-queues environment, is not defined to this IMS, and therefore, cannot be scheduled in this IMS subsystem.

IOPREV
Indicates that a BMP program containing GSAM cannot complete scheduling because I/O prevention has not completed. Further I/O requests to data sets are inhibited.

LCK
Transaction is locked by a /LOCK TRAN command.

NOTINIT
Transaction is not initialized and cannot be used.

QERR
I/O error has occurred on this queue for this MSC remote transaction.

QSTP
Transaction queuing is stopped by online change because the transaction is affected by the online change. Online change might be changing or deleting the transaction, or changing or deleting a program, PSB, database, or DMB referenced by the transaction. Transaction queuing is stopped until the online change is committed or aborted.

SUSPEND
Transaction has messages on the suspend queue.

STOQ
Transaction is stopped for queuing and can no longer be queued globally. This status might be caused by a previous UPDATE TRAN, /PURGE TRAN or /STO TRAN command.

STOSCHD
Transaction is stopped for scheduling and can no longer be scheduled

globally. This status might be caused by a previous UPDATE TRAN, /PSTOP TRAN or /STO TRAN command or an application abend.

TRACE
Transaction is being traced.

USTO
Transaction scheduling stopped because of unavailable data.

Usage notes

The transaction information displayed depends on whether the IMS issuing the QUERY TRAN command is running with RM services. If the QUERY TRAN command is issued by an IMS command master running without RM, all transaction information local to that IMS is returned. If the QUERY TRAN command is issued by an IMS command master running with RM, the IMS command master retrieves global information from CQS or RM as specified.

If the QUERY TRAN command is routed for global information to all IMS systems in an environment where some IMS systems use RM services and other IMS systems do not, the command results will vary because any of the IMS systems can be the command master. The RM environment of the IMS command master affects the type of transaction information that is displayed. You might want to route QUERY TRAN to specific IMS systems if some IMS systems are using RM. Here are two examples of why you might receive different command results:

1. The IMSplex has non-cloned systems and the transaction is only defined by an IMS that has RMENV=N. The IMS command master is running with RM services, but because the transaction is not defined to RM, no global information is obtained. The results are two response lines:
   - The IMS command master returns the transaction name as invalid.
   - The IMS with RMENV=N returns its local information.
2. The IMSplex has non-cloned systems. If global queue counts are requested, and the IMS command master does not have RM running, no global queue counts are returned for transactions that are not defined locally at the command master. All other IMS systems return only their local information.

In a shared-queues environment, a QUERY TRAN NAME(*tranname*) QCNT(GT,0) command returns the total count of messages on the shared queue structure for transaction *tranname*. This count includes messages that can be processed by any IMS (messages with no affinity), plus messages with affinity to an IMS. A QUERY TRAN NAME(*tranname*) QCNT(GT,0) SHOW(AFFIN) command returns one line for the transaction name with messages that can be processed by any IMS (messages with no affinity). Another line is returned for the transaction that has messages queued with affinity to an IMS. This line shows the number of messages that are queued with affinity, as well as the IMSID of the IMS to which they have affinity. A QUERY TRAN NAME(*tranname*) QCNT(GT,0) or QUERY TRAN NAME(*tranname*) QCNT(GT,0) SHOW(AFFIN) command is processed only by the command master IMS because the command master IMS can obtain information for all IMS subsystems in the IMSplex environment.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY TRAN command and the type-1 IMS commands that display similar information.

Table 138. Type-1 equivalents for the QUERY TRAN command

QUERY TRAN command	Similar IMS type-1 command
QUERY TRAN NAME(tranname) SHOW(ALL)	/DISPLAY TRAN tranname
QUERY TRAN SHOW(ALL)	/DISPLAY TRAN ALL
QUERY TRAN NAME(tranname) SHOW(QCNT)	/DISPLAY TRAN tranname QCNT
QUERY TRAN NAME(tranname) STATUS(IOPREV,LCK,QERR,SUSPEND,STOQ, STOSCHD,USTO) SHOW(STATUS)	/DISPLAY STATUS TRANSACTION

Output fields

The following table shows the output fields for the QUERY TRAN command. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that can appear for a nonzero completion code.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 139. Output fields for the QUERY TRAN command

Short label	Keyword	Scope	Meaning
AFIN	AFFIN	GBL	Affinity of the transaction messages on the shared queues, or affinity registration of the transactions for this IMS. For a transaction message on the shared queues (that is, the QCNT option), AFFIN displays the IMS ID or the recoverable service element (RSE) name of the IMS system that the message can be processed on. For a transaction registration, AFFIN displays the IMS ID or the RSE name appended to the transaction that was registered to CQS with local affinity.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
AOCMD	AOCMD	LCL	Transaction supports AOI CMD calls (CMD, TRAN, or Y) or not (N). **N** Indicates that the transaction is not permitted to issue AOI type-1 CMD calls. The transaction is permitted to issue AOI type-2 ICMD calls. **CMD** Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, authorization checking is based on which transactions can issue a particular command. In this case, the commands (or first three characters of the commands) need to be defined to RACF or equivalent product as a user. The type-1 AOI transactions must defined as profiles under the TIMS class, and for each transaction, the commands it can issue must be specified. Defining AOCMD(CMD) requires you to create fewer user IDs than you need to create for the AOCMD(TRAN) definition. However, defining AOCMD(CMD) requires you to create or modify a larger number of resource profiles. **TRAN** Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the transaction code is used for authorization. The first authorization check results in the accessor environment element (ACEE) being built. This environment is kept for use by future authorization checks. The type-1 AOI transaction need to be defined to RACF or equivalent product as a user. The transactions will then be specified on RACF PERMIT statements for each command they are allowed to issue from a type-1 AOI transaction. Specifying AOI transactions as users to RACF might conflict with the name of a user already defined to RACF. If this occurs, then either the transaction name or the existing user name needs to be changed. **Y** Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the user ID or program name is used for authorization. For some environments, if a Get Unique call has not yet occurred, the program name is used for authorization.
CC	N/A	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned. Refer to Table 143 on page 312 for more information.
CCTXT	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
CMTM	CMTMODE	LCL	Commit mode for the transaction: commit after a single message (SNGL) or multiple messages (MULT). **MULT** Database updates and non-express output messages are committed only when the application program terminates normally, when the processing limit count has been reached, or, in the case of a pseudo-WFI dependent region, when there are no more messages on the queue. For example, if five transactions are processed during a single schedule of a program, all five are committed only when the fifth one is completed and the program terminates. Until a transaction has been committed, locks for updated database records are not released and non-express output messages are not queued for output. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing. **SNGL** Database updates and non-express output messages are committed when the application program completes processing each transaction. IMS invokes commit processing either when the application program requests the next message (issues a GU to the IO-PCB), or when the application program terminates. If an application ends abnormally before committing its message, emergency restart requeues the message that was in process before the abend and makes it available for reprocessing.
CONV	CONV	LCL	Conversation option. Transaction is conversational (Y), or not (N). **N** Transaction is not conversational. **Y** Transaction is conversational. The transaction message is destined for a conversational program. A conversational program processes transactions made up of several steps. A conversational program receives a message from a terminal, replies to the terminal, but saves the data from the transaction in a scratchpad area (SPA). When the person at the terminal enters more data, the program has the data it saved from the last message in the SPA, so it can continue processing the request without the person at the terminal having to enter the data again. **Restriction:** The method to filter on a conversational transaction has changed between IMS Version 9 and IMS Version 10. In IMS Version 9, the command is QUERY TRAN STATUS(CONV). In IMS Version 10 or later, the command is QUERY TRAN CONV(Y). If IMS is running in an IMSplex with IMS Version 10 or later and either IMS Version 8 or IMS Version 9 systems, and the command is to be processed on all systems, the command must be issued as QUERY TRAN STATUS(CONV) CONV(Y). The IMS Version 10 system or later processes the CONV(Y) filter, while the IMS Version 8 and IMS Version 9 systems process the STATUS(CONV) filter. If a command is issued to an IMS Version 10 system or later with STATUS(CONV) without CONV(Y), the command is rejected.
CONVID	WORK	LCL	Conversation id for transaction that has a conversation in progress.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
DCLW	DCLWA	LCL	Perform log write-ahead for recoverable, nonresponse mode input messages and transaction output messages (Y) or not (N). **N** IMS does not perform log write-ahead. **Y** IMS performs log write-ahead for recoverable, nonresponse input messages and transaction output messages. If not defined for the transaction, the default is the DCLWA parameter in the IMSCTRL macro. This ensures that a nonresponse input transaction is made recoverable across IMS failures, prior to IMS acknowledging receipt of the input. Database changes are made recoverable prior to IMS sending associated output reply messages. This ensures that information in the log buffers is written to the IMS log, before the associated input acknowledgment or output reply is sent to the terminal.
DFNT	DEFNTYPE	LCL	Definition type, which can be one of the following: **CPIC** A CPI-C transaction that was dynamically built on this IMS system. **CREATE** Defined by a CREATE command. **DFSINSX0** Defined by user exit DFSINSX0. The resource can only be exported if the export option was set. This does not apply to descriptors. **IMPORT** Defined by an IMPORT command. **IMS** Defined by IMS. **MODBLKS** Defined by system definition in the MODBLKS data set. **UPDATE** Defined by system definition in the MODBLKS data set, but changed into a dynamic resource by an UPDATE command. This does not apply to descriptors.
DRRT	DIRROUTE	LCL	Supports MSC directed routing (Y) or not (N). **N** The application program processing a transaction is not informed of the system which originated the transaction. The name of the originating LTERM is placed in the I/O PCB. **Y** The application program processing a transaction is informed of the system which originated the transaction, if MSC directed routing is used in a multiple IMS system configuration. An MSNAME corresponding to a logical path back to the originating system is placed in the I/O PCB.
EDTR	EDITRTN	LCL	Input edit routine name.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
EDTT	EDITUC	LCL	Input data is to be translated to uppercase (Y) or not (N). **N** Input data is not translated to uppercase. It can consist of uppercase and lowercase characters as entered from the terminal. **Y** Input data is to be translated to uppercase before it is presented to the processing program. If FP(E), the transaction is to be translated to uppercase before being presented to the edit/routing exit routine. Specifying EDITUC(Y) for VTAM terminals prevents the transmission of embedded device control characters.
EMHBS	EMHBSZ	LCL	EMH buffer size.
EXPRT	EXPRTIME	LCL	Transaction expiration time
FP	FP	LCL	Fast Path potential candidate (P), Fast Path exclusive (E), or FP option not enabled (N). **E** Fast Path exclusive transaction. Any message for this transaction is always routed to a Fast Path application program. **N** Fast Path option is not enabled. When FP(N) is specified, any attempt to use Fast Path resources or commands will yield unpredictable results. **P** Fast Path potential transaction. Any message for this transaction can potentially be routed to a Fast Path application program. **Restriction:** The method to filter on a Fast Path transaction has changed between IMS Version 9 and IMS Version 10. In Version 9, the command is QUERY TRAN STATUS(FPE), QUERY TRAN STATUS(FPP) or QUERY TRAN STATUS(FPE,FPP). In IMS Version 10 or later, the command is QUERY TRAN FP(E), QUERY TRAN FP(P), or QUERY TRAN FP(E,P). If IMS is running in an IMSplex with IMS Version 10 or later and either IMS Version 8 or IMS Version 9 systems, and the command is to be processed on all systems, the command must be issued as QUERY TRAN STATUS(FPE) FP(E), QUERY TRAN STATUS(FPP) FP(P), or QUERY TRAN STATUS(FPE,FPP) FP(E,P). The IMS Version 10 system or later processes the FP() filter, while the IMS Version 8 and IMS Version 9 systems process the STATUS() filter. If a command is issued to an IMS Version 10 system or later with STATUS(FPE), STATUS(FPP), or STATUS(FPE,FPP) without FP(), the command is rejected.
INQ	INQ	LCL	Inquiry transaction (Y) or not (N). **N** Inquiry option is disabled. **Y** Inquiry option is enabled. This is an inquiry transaction that, when entered, does not cause a change in any database. Programs are prohibited from issuing ISRT, DLET, or REPL calls to a database when scheduled to process a transaction defined as INQ(Y). An application program cannot do an SQL INSERT, DELETE, or UPDATE when the IMS transaction is defined with INQ(Y).
LCLS	CLASS	LCL	Scheduling class used to determine which message regions can process the transaction locally on a particular IMS.
LCP	CPRI	LCL	Local current scheduling priority. The current scheduling priority is used to calculate which transaction is selected for scheduling.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
LLCT	LCT	LCL	Limit count in the local IMS. The limit count is the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction.
LLP	LPRI	LCL	Local limit scheduling priority. The limit scheduling priority is the priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value.
LMRG	MAXRGN	LCL	Local maximum region count. The maximum region count is the maximum number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction that is eligible for parallel scheduling.
LNP	NPRI	LCL	Local normal scheduling priority. The normal scheduling priority is the priority assigned to this transaction when the number of input transactions enqueued and waiting to be processed is less than the limit count value.
LPLCT	PLCT	LCL	Local processing limit count. The processing limit count is the number of transaction messages a program can process in a single scheduling.
LPLM	PARLIM	LCL	Local parallel processing limit count. The parallel limit count is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. An additional message region is scheduled whenever the transaction queue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.
LQ	QCNT	LCL	Local transaction message queue count.
LSNO	SEGNO	LCL	Local application program output segment limit allowed in message queues for each GU call.
LSSZ	SEGSZ	LCL	Local application program output segment size limit allowed in the message queues for each GU call.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
LSTT	STATUS	LCL	Local transaction status. **BAL** Transaction is eligible for load balancing (for example, with parallel limits specified). **CPIC** This CPI-C transaction was built dynamically on this IMS system and can process only on this IMS system. **DYN** Transaction was built in a shared-queues environment, is not defined to this IMS, and therefore, cannot be scheduled in this IMS subsystem. **IOPREV** Indicates that a BMP program containing GSAM cannot completed scheduling because I/O prevention has not completed. Further I/O requests to data sets are inhibited. **LOCK** Transaction is locked by a /LOCK TRAN or UPD TRAN SET(LOCK(ON)) command. **NOTINIT-***xx-reason* Transaction is not initialized and cannot be used. NOTINIT status is displayed in the format NOTINIT-*xx-reason*, where *xx* is the reason code that identifies the unique location in one module where this reason code is set. NOTINIT-00 indicates that the reason is unknown. Action: 1. IAPS MACRO defines each reason code that may be set in the transaction bad reason code (field SMBBADR) and identifies the module that sets it. *reason* explains the reason code *xx* in abbreviated text format up to 13 characters. For possible *reason* values and their descriptions, see Table 140 on page 309. **QERR** I/O error has occurred on this queue for this MSC remote transaction. **QSTP** Transaction queuing is stopped by online change because the transaction is affected by the online change. Online change might be changing or deleting the transaction, or changing or deleting a program, PSB, database, or DMB referenced by the transaction. Transaction queuing is stopped until the online change is committed or aborted. **SUSPEND** Transaction has messages on the suspend queue. **STOQ** Transaction is stopped for queuing, and messages can no longer be queued to it. This status might be caused by a previous UPDATE TRAN, /PURGE TRAN, or /STO TRAN command. **STOSCHD** Transaction is stopped for scheduling and can no longer be scheduled. This status might be caused by a previous UPDATE TRAN, /PSTOP TRAN, or /STO TRAN command or an application abend. **TRACE** Transaction is being traced. **USTO** Transaction scheduling stopped because of unavailable data.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
LU	WORK	LCL	APPC Luname that initiated conversation.
MDLN	MODEL	LCL	Model name. Name of the resource used as a model to create this resource. DFSDSTR1 is the IMS descriptor name for transactions.
MDLT	MODEL	LCL	Model type, either RSC or DESC. RSC means the resource was created using another resource as a model. DESC means the resource was created using a descriptor as a model.
MBR	N/A	N/A	IMSplex member that built the output line. IMS identifier of IMS that built the output. The IMS identifier is always returned.
MSGT	MSGTYPE	LCL	Message type of single segment (SNGLSEG) or multiple segment (MULTSEG). **MULTSEG** Specifies that the incoming message can be more than one segment in length. It is not eligible for scheduling to an application program until an end-of-message indication is received, or a complete message is created by MFS. **SNGLSEG** Specifies that the incoming message is one segment in length. It becomes eligible for scheduling when the terminal operator indicates end-of-segment.
MSN	MSNAME	LCL	Logical link path name.
NODE	WORK	LCL	Node name that initiated conversation.
PLCTT	PLCTTIME	LCL	Processing limit count time.
PSB	PSB	LCL	PSB name associated with the transaction.
Q	QCNT, GLOBAL	GBL	Global transaction message queue count on the shared queues. Q is only displayed if shared queues are used.
RCV	RECOVER	LCL	Recovered during an IMS emergency or normal restart (Y) or not (N). **N** Recovery option is disabled. The transaction is not recovered. **Y** Recovery option is enabled. The transaction is recovered during IMS emergency or normal restart.
RGC	RGC	LCL	Number of regions the transaction is currently scheduled in the local IMS.
RMT	REMOTE	LCL	Remote transaction (Y) or not (N). **N** Local transaction. The transaction runs on the local system. **Y** Remote transaction. The transaction runs on a remote system. **Restriction:** The method to filter on a remote transaction changed between IMS Version 9 and IMS Version 10. In Version 9, the command is QUERY TRAN STATUS(RMT). In IMS Version 10 or later, the command is QUERY TRAN REMOTE(Y). If IMS is running in an IMSplex with IMS Version 10 or later and either IMS Version 8 or IMS Version 9 systems, and the command is to be processed on all systems, the command must be issued as QUERY TRAN STATUS(RMT) REMOTE(Y). The IMS Version 10 system or later processes the REMOTE(Y) filter, while the IMS Version 8 and IMS Version 9 systems process the STATUS(RMT) filter. If a command is issued to an IMS Version 10 system or later with STATUS(RMT) without REMOTE(Y), the command is rejected.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
RSP	RESP	LCL	Response mode transaction (Y) or not (N). **N** Response mode option is disabled. For terminals specifying or accepting a default of OPTIONS=TRANRESP, input should not stop after this transaction is entered. **Y** Response mode option is enabled. For terminals specifying or accepting a default of OPTIONS=TRANRESP, no additional messages are to be allowed after this transaction is entered until this transaction sends a response message back to the terminal. Response mode can be forced or negated by individual terminal definition. RESP(Y) is ignored during online processing for all terminals that do no operate in response mode. **Restriction:** The method to filter on a remote transaction changed between IMS Version 9 and IMS Version 10. In Version 9, the command is QUERY TRAN STATUS(RESP). In IMS Version 10 or later, the command is QUERY TRAN RESP(Y). If IMS is running in an IMSplex with IMS Version 10 or later and either IMS Version 8 or IMS Version 9 systems, and the command is to be processed on all systems, the command must be issued as QUERY TRAN STATUS(RESP) RESP(Y). The IMS Version 10 system or later processes the RESP(Y) filter, while the IMS Version 8 and IMS Version 9 systems process the STATUS(RESP) filter. If a command is issued to an IMS Version 10 system or later with STATUS(RESP) without RESP(Y), the command is rejected.
SER	SERIAL	LCL	Transaction is processed serially (Y) or not (N). **N** Serial option is disabled. Messages for the transaction are not processed serially. Message processing can be processed in parallel. Messages are placed on the suspend queue after a U3303 pseudoabend. Scheduling continues until repeated failures result in the transaction being stopped with a USTOP. **Y** Serial option is enabled. Messages for the transaction are processed serially. U3303 pseudoabends do not cause the message to be placed on the suspend queue but rather on the front of the transaction message queue, and the transaction is stopped with a USTOP. The USTOP of the transaction is removed when the transaction or the class is started with a /START command.
SIDL	MSNAME	LCL	Local system ID.
SIDR	MSNAME	LCL	Remote system ID.
SPASZ	SPASZ	LCL	Conversational transaction scratchpad area size.
SPATR	SPATRUNC	LCL	Conversational transaction SPA data should be truncated (R) or preserved (S) across a program switch to a transaction that is defined with a smaller SPA. **S** IMS preserves all of the data in the SPA, even when a program switch is made to a transaction that is defined with a smaller SPA. The transaction with the smaller SPA does not see the truncated data, but when the transaction switches to a transaction with a larger SPA, the truncated data is used. **R** Truncated data is not preserved.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
STT	GLOBAL, STATUS	GBL	Global transaction status, which can be one of the following: **STAQ** Transaction has a global status of started for queuing. **STASCHD** Transaction has a global status of started for scheduling. **STOQ** Transaction has a global status of stopped for queuing. **STOSCHD** Transaction has a global status of stopped for scheduling.
TLS	TRANSTAT	LCL	Transaction level statistics logged (Y) or not (N). **N** Transaction level statistics logging is not active. **Y** Transaction level statistics logging is active.
TMAC	TIMESTAMP	LCL	The time that the resource was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. The transaction time stamp is not updated for Fast Path exclusive transactions; only the associated routing code time stamp is updated. For a transaction descriptor, the following actions update the last access time: • Message is queued to the transaction from any source (program, command, other). • Message is dequeued by an application program (MSG GU). If a DEQ command is issued to dequeue a message, that action does not update the access time. • CREATE command or DFSINSX0 exit references the resource as a model.
TMCR	TIMESTAMP	LCL	The time that the resource was created. This is the result of a CREATE TRAN command, IMPORT command that creates the transaction, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMEM	WORK	LCL	OTMA tmember that initiated conversation.
TMIM	TIMESTAMP	LCL	The time that the resource was last imported. The import time is retained across warm start and emergency restart.
TMUP	TIMESTAMP	LCL	The time that the resource was updated as a result of an UPDATE TRAN command that changes a runtime resource definition attribute. This includes attributes set by the SET keyword. The update time is retained across warm start, emergency restart, EXPORT and IMPORT.
TPIP	WORK	LCL	OTMA tpipe that initiated conversation.
TRAN	TRAN	LCL	Transaction name. A transaction defines the processing characteristics of messages destined for an application program.
USER	WORK	LCL	User that initiated conversation.

Table 139. Output fields for the QUERY TRAN command (continued)

Short label	Keyword	Scope	Meaning
WFI	WFI	LCL	Wait-for-input transaction (Y) or not (N). **N** Wait-for-input option is disabled. **Y** Wait-for-input option is enabled. A message processing or batch processing application program that processes WFI transactions is scheduled and invoked normally. If the transaction to be processed is defined as WFI, the program is allowed to remain in main storage after it has processed the available input messages. The QC status code (no more messages) is returned to the program if the processing limit count is reached; a command is entered to change the status of the schedule transaction, database, program or class; commands relating to the databases used by the transaction are entered; or IMS is terminated with a checkpoint shutdown.
WRK	WORK	LCL	Work is in progress for the transaction or one of its associated resources. The work in progress can be one of the following: ANOTHER CMD IN PROGRESS Another command (DELETE, IMPORT, UPDATE) to delete or update the transaction is already in progress. IN CONVERSATION Transaction is in conversation. Additional information that uniquely identifies the conversation such as the conversation ID, username or ltermname (if conversation stored in RM), node, lterm, user, luname, tmember, or tpipe is returned as separate information with unique XML tags. IN USE Transaction is in use. Queuing is in progress, either terminal input or a program-to-program switch. QUEUEING Transaction has messages queued in a non-shared-queues environment. SCHEDULED Transaction is scheduled or application program referenced by transaction is scheduled. SUSPENDED Transaction is on the suspend queue.

Table 140. Reason information for NOTINIT-xx-reason status

Reason	Meaning
EDTRTN	The input edit routine that is referred to by the transaction could not be loaded.
NOMSN	The MSNAME that is referred to by the transaction does not exist.
NOPGM	The program that this transaction references is not defined. No program PDIR control block exists. Action: 2.

Table 140. Reason information for NOTINIT-xx-reason status (continued)

Reason	Meaning
NORTC	The Fast Path routing code that this transaction references is not defined. No Fast Path routing code RCTE control block exists. Action: 3.
NOTBL	The table that is used to manage transaction input edit routines could not be allocated.

Note: Actions that can be taken to initialize the transaction are:

1. Call IBM.
2. Issue a CREATE PGM command to create the program. Issue an UPDATE TRAN START(Q,SCHD) command to complete transaction initialization.
3. Issue a CREATE RTC command to create the routing code. Issue an UPDATE TRAN START(Q,SCHD) command to complete transaction initialization.

How the SHOW keyword on QUERY TRAN determines the type of output

The following table provides some examples of how the SHOW keyword determines the type of output returned on the QUERY TRAN command.

Table 141. How the SHOW keyword on QUERY TRAN determines the type of output

Form of SHOW keyword used	Type of output returned
SHOW(LOCAL)	Only those output fields that are local to an IMS system. SHOW(ALL,LOCAL) provides the same output.
SHOW(GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM or CQS. SHOW(ALL,GLOBAL) provides the same output.
SHOW(ALL)	All the output fields for those fields that have both local and global data. Both values are returned in the output.
SHOW(QCNT,GLOBAL)	Only global QCNT values.
SHOW(QCNT,LOCAL)	Only local QCNT values.
SHOW(QCNT)	Both local and global QCNT values, because there is no LOCAL or GLOBAL qualifier.
SHOW(ALL,GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM or CQS. SHOW(GLOBAL) provides the same output.
SHOW(ALL,LOCAL)	Only those output fields that are local to an IMS system. SHOW(LOCAL) provides the same output.

If specific output fields are listed in the SHOW keyword, you can specify them as either local or global; however, you cannot combine a list of SHOW keywords in which some are global and some are local.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY TRAN command are standard for all commands entered through the OM API.

The following table contains the return and reason codes that can be returned to OM from a QUERY TRAN command.

Table 142. Return and reason codes for the QUERY TRAN command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY TRAN command completed successfully.
X'00000004'	X'00000010'	No resources were found to be returned. The resource names specified might be invalid, or there were no resources that match the filter specified, or there were no resources that had work to display for the SHOW(WORK) specified.
X'00000004'	X'00001000'	The QUERY TRAN command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000004'	X'00001004'	The QUERY TRAN command was processed for a few resources and a partial list of resources is returned. The command terminated as the resource count to be returned exceeded the maximum number of resources that can be returned by a QUERY TRAN command. The maximum number of resources that can be returned by a QUERY TRAN command is 5000. Issue the command again with a generic mask or other filters so the number of resources to be returned is less than 5000.
X'00000004'	X'00001010'	No resources were found to be returned. The resource names specified might be invalid or there were no resources that match the filter specified.
X'00000008'	X'00002040'	The QUERY TRAN command has more than one filter value specified or an invalid filter value is specified for the QCNT filter. Only one of the CLASS, STATUS, or QCNT filters can be specified. A value of 0 might have been specified for QCNT with LE, GE or, EQ. Or a value of 1 might have been specified for QCNT with LT.
X'00000008'	X'0000204C'	The CLASS value specified is invalid. Confirm the correct CLASS value is specified on the command.
X'00000008'	X'00002137'	Transaction referenced as a model is a queue-only transaction with a status of DYN, which cannot be used as a model.
X'0000000C'	X'00003000'	The QUERY TRAN command is successful for at least one resource name. The QUERY TRAN command is not successful for one or more resource names. The completion code indicates the reason for the error with the resource name.
X'0000000C'	X'00003004'	The QUERY TRAN command is not successful for all the resource names specified. The completion code indicates the reason for the error with the resource name.
X'00000010'	X'00004014'	The QUERY TRAN command processing terminated as the TRAN keyword is not valid on the RSR tracker.

Table 142. Return and reason codes for the QUERY TRAN command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004018'	The QUERY TRAN command processing terminated as the RM resource structure is not available.
X'00000010'	X'00004100'	The QUERY TRAN command processing terminated as the resource structure is full.
X'00000010'	X'00004104'	The QUERY TRAN command processing terminated as there is no RM address space.
X'00000010'	X'00004108'	The QUERY TRAN command processing terminated as there is no SCI address space.
X'00000014'	X'00005004'	The QUERY TRAN command processing terminated as a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	The QUERY TRAN command processing terminated as the DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	The QUERY TRAN command processing terminated because of a RM error.
X'00000014'	X'00005108'	The QUERY TRAN command processing terminated because of a SCI error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains completion codes that can be returned on a QUERY TRAN command.

Table 143. Completion codes for the QUERY TRAN command

Completion code	Completion code text	Meaning
0		Command completed successfully for transaction.
10	NO RESOURCES FOUND	Transaction name is invalid, or the wildcard parameter specified does not match any resource names.
50	CQS IS UNAVAILABLE	The QUERY TRAN command could not be completed for the resource as CQS is not available. Make sure CQS is available before issuing the command again.
51	NO RESOURCE STRUCTURE	The QUERY TRAN command could not be completed for the resource as there is no resource structure or it is unavailable.
52	RESOURCE STRUCTURE FULL	The QUERY TRAN command could not be completed for the resource as the resource structure is full.
90	INTERNAL ERROR	The QUERY TRAN command could not be completed for the resource because of an IMS internal error.
94	RM REQUEST ERROR	The QUERY TRAN command could not be completed for the resource because of an RM error.

Table 143. Completion codes for the QUERY TRAN command (continued)

Completion code	Completion code text	Meaning
98	CQS REQUEST ERROR	The QUERY TRAN command could not be completed for the resource because of a CQS error.

Examples

The following are examples of the QUERY TRAN command:

Example 1 for QUERY TRAN command

TSO SPOC input:

```
QRY TRAN NAME(PART,TRAN000%,CDEBTRN4,CONV13V0,TXBANKI4,FPACP) SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
Trancode MbrName    CC PSBname  LCls    LQCnt  LLCT LPLCT LPLCTTime LCPRI LNPRI
CDEBTRN4 IMS1        0 CDEBS       1        0 65535 65535   6553500     1     1
CONV13V0 IMS1        0 CPGM1V0     1        0 65535 65535   6553500     1     1
FPACP    IMS1        0 TACP1       1        0 65535 65535   6553500     1     1
PART     IMS1        0 DFSSAM02    4        0     2 65535   6553500     7     7
TRAN0001 IMS1        0 DFSSAM02    4        0     2 65535   6553500     7     7
TRAN0002 IMS1        0 DFSSAM04    1        0 65535 65535   6553500     1     1
TXBANKI4 IMS1        0 BANKIFP     1        0     0 65535   6553500     1     1
(scrolled to the right screen 2)
Trancode MbrName  LLPRI LSegSz LSegNo LParLim RegCnt LMaxRgn EditRtn  FP EMHBSz
CDEBTRN4 IMS1         1      0      0   65535      0       0          N       0
CONV13V0 IMS1         1      0      0   65535      0       0          N       0
FPACP    IMS1         1      0      0   65535      0       0          P       0
PART     IMS1        10      0      0   65535      0       0          N       0
TRAN0001 IMS1        10      0      0   65535      0       0          N       0
TRAN0002 IMS1         1      0      0   65535      0       0          N       0
TXBANKI4 IMS1         1      0      0   65535      0       0          E       0
(scrolled to the right screen 3)
Trancode MbrName  CmtMode MsgType SPATrunc SPASz  SIDR  SIDL DCLWA DirRoute Edit
CDEBTRN4 IMS1     SNGL    MULTSEG        S  1000    10    10 Y     N        Y
CONV13V0 IMS1     SNGL    MULTSEG        S  1000    16    36 Y     N        Y
FPACP    IMS1     SNGL    SNGLSEG           0    10    10 Y     N        Y
PART     IMS1     MULT    MULTSEG           0    10    10 Y     N        Y
TRAN0001 IMS1     MULT    MULTSEG           0    10    10 Y     N        Y
TRAN0002 IMS1     SNGL    MULTSEG           0    10    10 Y     N        Y
TXBANKI4 IMS1     SNGL    SNGLSEG           0    10    10 Y     N        Y
(scrolled to the right screen 4)
Trancode MbrName  tUC Inq Recover Resp Remote Serial WFI AOCMD Conv TranStat Lcl
CDEBTRN4 IMS1         N   Y       N    N      N      N   N     Y    N
CONV13V0 IMS1         N   Y       N    Y      N      N   N     Y    N
FPACP    IMS1         N   Y       Y    N      N      N   N     N    N
PART     IMS1         Y   Y       N    N      N      N   N     N    N        STO
TRAN0001 IMS1         Y   Y       N    N      N      N   N     N    N
TRAN0002 IMS1         N   Y       N    N      N      N   N     N    N
TXBANKI4 IMS1         N   Y       Y    N      N      N   N     N    N
(scrolled to the right screen 5)
Trancode MbrName  <LclStat  ModelName ModelType MSName   TimeAccess           Ti
CDEBTRN4 IMS1
CONV13V0 IMS1                                   LINK31V1
FPACP    IMS1
PART     IMS1     Q,STOSCHD                             2006.286 13:48:46.66
TRAN0001 IMS1               PART      RSC
TRAN0002 IMS1               DFSDSTR1  DESC
TXBANKI4 IMS1
```

```
(scrolled to the right screen 6)
Trancode MbrName  <TimeUpdate        TimeCreate           TimeImport
CDEBTRN4 IMS1                       2006.286 09:14:54.04
CONV13V0 IMS1                       2006.286 09:14:54.04
FPACP    IMS1                       2006.286 09:14:54.04
PART     IMS1                       2006.286 09:14:54.04
TRAN0001 IMS1                       2006.286 13:48:46.66
TRAN0002 IMS1                       2006.286 13:49:25.86
TXBANKI4 IMS1                       2006.286 09:14:54.04
(scrolled to the right screen 7)
Trancode MbrName  TimeCreate           TimeImport           DefnType   EXPRTm
CDEBTRN4 IMS1     2006.286 09:14:54.04                      MODBLKS       500
CONV13V0 IMS1     2006.286 09:14:54.04                      MODBLKS       500
FPACP    IMS1     2006.286 09:14:54.04                      MODBLKS       200
PART     IMS1     2006.286 09:14:54.04                      MODBLKS       100
TRAN0001 IMS1     2006.286 13:48:46.66                      CREATE        450
TRAN0002 IMS1     2006.286 13:49:25.86                      CREATE        550
TXBANKI4 IMS1     2006.286 09:14:54.04                      MODBLKS       300
```

OM API input:

```
CMD(QUERY TRAN NAME(PART,TRAN000%,CDEBTRN4,CONV13V0,TXBANKI4,FPACP) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.286 20:55:09.384224</statime>
<stotime>2006.286 20:55:09.384867</stotime>
<staseq>BF8C7C9AA6220487</staseq>
<stoseq>BF8C7C9AA64A3147</stoseq>
<rqsttkn1>USRT011 10135509</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>TRAN            </kwd>
<input>QRY TRAN NAME(PART,TRAN000%,CDEBTRN4,CONV13V0,TXBANKI4,FPACP)
 SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="CONVID" llbl="ConvID" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="NODE" llbl="NodeName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="USER" llbl="UserName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="LU" llbl="LUName" scope="LCL" sort="n" key="0"
 scroll="yes"len="24" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMEM" llbl="TMember" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TPIP" llbl="TPipe" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="PSB" llbl="PSBname" scope="LCL" sort="n" key="0"
```

```
                        scroll="yes" len="8" dtype="CHAR" align="left" />
                       <hdr slbl="LCLS" llbl="LCls" scope="LCL" sort="n" key="0" scroll="yes"
                        len="3" dtype="INT" align="right" />
                       <hdr slbl="LQ" llbl="LQCnt" scope="LCL" sort="d" key="4" scroll="yes"
                        len="8" dtype="INT" align="right" />
                       <hdr slbl="LLCT" llbl="LLCT" scope="LCL" sort="n" key="0" scroll="yes"
                        len="5" dtype="INT" align="right" />
                       <hdr slbl="LPLCT" llbl="LPLCT" scope="LCL" sort="n" key="0"
                        scroll="yes" len="5" dtype="INT" align="right" />
                       <hdr slbl="PLCTT" llbl="LPLCTTime" scope="LCL" sort="n" key="0"
                       scroll="yes" len="7" dtype="INT" align="right" />
                       <hdr slbl="LCP" llbl="LCPRI" scope="LCL" sort="n" key="0" scroll="yes"
                        len="2" dtype="INT" align="right " />
                       <hdr slbl="LNP" llbl="LNPRI" scope="LCL" sort="n" key="0" scroll="yes"
                        len="2" dtype="INT" align="right" />
                       <hdr slbl="LLP" llbl="LLPRI" scope="LCL" sort="n" key="0" scroll="yes"
                        len="2" dtype="INT" align="right" />
                       <hdr slbl="LSSZ" llbl="LSegSz" scope="LCL" sort="n" key="0"
                        scroll="yes" len="5" dtype="INT" align="right" />
                       <hdr slbl="LSNO" llbl="LSegNo" scope="LCL" sort="n" key="0"
                        scroll="yes" len="5" dtype="INT" align="right" />
                       <hdr slbl="LPLM" llbl="LParLim" scope="LCL" sort="n" key="0"
                        scroll="yes" len="5" dtype="INT" align="right" />
                       <hdr slbl="RGC" llbl="RegCnt" scope="LCL" sort="n" key="0" scroll="yes"
                        len="5" dtype="INT" align="right" />
                       <hdr slbl="LMRG" llbl="LMaxRgn" scope="LCL" sort="n" key="0"
                        scroll="yes" len="5" dtype="INT" align="right" />
                       <hdr slbl="EDTR" llbl="EditRtn" scope="LCL" sort="n" key="0"
                        scroll="yes" len="8" dtype="CHAR" align="left" />
                       <hdr slbl="FP" llbl="FP" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" />
                       <hdr slbl="EMHBS" llbl="EMHBSz" scope="LCL" sort="n" key="0"
                        scroll="yes" len="5" dtype="INT" align="right" />
                       <hdr slbl="CMTM" llbl="CmtMode" scope="LCL" sort="n" key="0"
                        scroll="yes" len="4" dtype="CHAR" align="left" />
                       <hdr slbl="MSGT" llbl="MsgType" scope="LCL" sort="n" key="0"
                        scroll="yes" len="7" dtype="CHAR" align="left" />
                       <hdr slbl="SPATR" llbl="SPATrunc" scope="LCL" sort="n" key="0"
                        scroll="yes" len="1" dtype="CHAR" align="right" />
                       <hdr slbl="SPASZ" llbl="SPASz" scope="LCL" sort="n" key="0"
                        scroll="yes" len="5" dtype="INT" align="right" />
                       <hdr slbl="SIDR" llbl="SIDR" scope="LCL" sort="n" key="0" scroll="yes"
                        len="5" dtype="INT" align="right" />
                       <hdr slbl="SIDL" llbl="SIDL" scope="LCL" sort="n" key="0" scroll="yes"
                        len="5" dtype="INT" align="right" />
                       <hdr slbl="DCLW" llbl="DCLWA" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" />
                       <hdr slbl="DRRT" llbl="DirRoute" scope="LCL" sort="n" key="0"
                        scroll="yes" len="1" dtype="CHAR" align="left" />
                       <hdr slbl="EDTT" llbl="EditUC" scope="LCL" sort="n" key="0"
                        scroll="yes" len="1" dtype="CHAR" align="left" />
                       <hdr slbl="INQ" llbl="Inq" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" />
                       <hdr slbl="RCV" llbl="Recover" scope="LCL" sort="n" key="0"
                        scroll="yes" len="1" dtype="CHAR" align="left" />
                       <hdr slbl="RSP" llbl="Resp" scope="LCL" sort="n" key="0" scroll="yes"
                       len="1" dtype="CHAR" align="left" />
                       <hdr slbl="RMT" llbl="Remote" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" />
                       <hdr slbl="SER" llbl="Serial" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" />
                       <hdr slbl="WFI" llbl="WFI" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" />
                       <hdr slbl="AOCMD" llbl="AOCMD" scope="LCL" sort="n" key="0"
                        scroll="yes" len="4" dtype="CHAR" align="left" />
                       <hdr slbl="CONV" llbl="Conv" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" />
```

```
<hdr slbl="TLS" llbl="TranStat" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" />
<hdr slbl="MDLN" llbl="ModelName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MDLT" llbl="ModelType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MSN" llbl="MSName" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="TMAC" llbl="TimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMUP" llbl="TimeUpdate" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMIM" llbl="TimeImport" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="DFNT" llbl="DefnType" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 </cmdrsphdr>
 <cmdrspdata>
 <rsp>TRAN(PART    ) MBR(IMS1    ) CC(   0)  PSB(DFSSAM02) LCLS(  4) LQ(
     0) LLCT(    2) LPLCT(65535) LCP( 7) LNP( 7) LLP(10) LSSZ(    0)
  LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) LSTT(STOQ,STOSCHD)
  AOCMD(N) CMTM(MULT) CONV(N) DCLW(Y) DFNT(MODBLKS) DRRT(N) EDTT(Y)
  EMHBS(    0) FP(N) INQ(Y) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N)
  RSP(N) SER(N) SIDL( 10) SIDR( 10) SPASZ(    0) TMAC(2006.286
  13:48:46.66) TMCR(2006.286 09:14:54.04) TLS(N) WFI(N)</rsp>
 <rsp>TRAN(TRAN0001) MBR(IMS1    ) CC(   0)  PSB(DFSSAM02) LCLS(  4) LQ(
     0) LLCT(    2) LPLCT(65535) LCP( 7) LNP( 7) LLP(10) LSSZ(    0)
  LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(MULT)
  CONV(N) DCLW(Y) DFNT(CREATE) DRRT(N) EDTT(Y) EMHBS(    0) FP(N) INQ(Y)
  MDLT(RSC) MDLN(PART    ) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N)
  RSP(N) SER(N) SIDL( 10) SIDR( 10) SPASZ(    0) TMCR(2006.286
13:48:46.66) TLS(N) WFI(N)</rsp>
<rsp>TRAN(TRAN0002) MBR(IMS1    ) CC(   0)  PSB(DFSSAM04) LCLS(  1) LQ(
    0) LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
 LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL)
 CONV(N) DCLW(Y) DFNT(CREATE) DRRT(N) EDTT(Y) EMHBS(    0) FP(N) INQ(N)
 MDLT(DESC) MDLN(DFSDSTR1) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N)
 RSP(N) SER(N) SIDL( 10) SIDR( 10) SPASZ(    0) TMCR(2006.286
 13:49:25.86) TLS(N) WFI(N)</rsp>
<rsp>TRAN(CDEBTRN4) MBR(IMS1    ) CC(   0)  PSB(CDEBS   ) LCLS(  1) LQ(
    0) LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
 LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL)
 CONV(Y) DCLW(Y) DFNT(MODBLKS) DRRT(N) EDTT(Y) EMHBS(    0) FP(N)
 INQ(N) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(N) SER(N) SIDL(
 10) SIDR( 10) SPASZ( 1000) SPATR(S) TMCR(2006.286 09:14:54.04) TLS(N)
 WFI(N)</rsp>
<rsp>TRAN(CONV13V0) MBR(IMS1    ) CC(   0)  PSB(CPGM1V0 ) LCLS(  1) LQ(
    0) LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
 LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL)
 CONV(Y) DCLW(Y) DFNT(MODBLKS) DRRT(N) EDTT(Y) EMHBS(    0) FP(N)
 INQ(N) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(Y) RSP(N) SER(N) SIDL(
 36) SIDR( 16) MSN(LINK31V1) SPASZ( 1000) SPATR(S) TMCR(2006.286
 09:14:54.04) TLS(N) WFI(N)</rsp>
<rsp>TRAN(TXBANKI4) MBR(IMS1    ) CC(   0)  PSB(BANKIFP ) LCLS(  1) LQ(
    0) LLCT(    0) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
 LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL)
 CONV(N) DCLW(Y) DFNT(MODBLKS) DRRT(N) EDTT(Y) EMHBS(    0) FP(E)
 INQ(N) MSGT(SNGLSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(Y) SER(N) SIDL(
 10) SIDR( 10) SPASZ(    0) TMCR(2006.286 09:14:54.04) TLS(N) WFI(N)
</rsp>
<rsp>TRAN(FPACP   ) MBR(IMS1    ) CC(   0)  PSB(TACP1   ) LCLS(  1) LQ(
    0) LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
 LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL)
```

```
 CONV(N) DCLW(Y) DFNT(MODBLKS) DRRT(N) EDTT(Y) EMHBS(    0) FP(P)
 INQ(N) MSGT(SNGLSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(Y) SER(N) SIDL(
 10) SIDR( 10) SPASZ(    0) TMCR(2006.286 09:14:54.04) TLS(N) WFI(N)
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY TRAN command is specified with SHOW(ALL) to display all the output fields for the specified transactions. All the transaction output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The transaction name and the member name that built the line of output are displayed on every screen. Transaction PART has a status of stopped for scheduling and queuing. Transactions CDEBTRN4 and CONV13V0 are conversational. Transaction FPACP is Fast Path potential. Transaction CONV13V0 is remote. Transactions TRAN0001 and TRAN0002 are dynamic, they display the ModelName and ModelType used to create them, and they have a definition type of CREATE.

Example 2 for QUERY TRAN command

TSO SPOC input:

```
QUERY TRAN STATUS(AFFIN)
```

TSO SPOC output:

```
 Trancode MbrName    CC LclStat
 APOL12   IMS1        0 AFFIN
```

Explanation: This example shows the use of the AFFIN option as a filter to query transactions that have affinity status.

Example 3 for QUERY TRAN command

TSO SPOC input:

```
QUERY TRAN NAME(APOL11 APOL12) SHOW(STATUS)
```

TSO SPOC output:

```
 Trancode MbrName    CC LclStat
 APOL11   IMS1        0
 APOL12   IMS1        0 AFFIN
```

Explanation: This example queries transaction status for transactions APOL11 and APOL12. The output shows that transaction APOL12 has affinity status.

Example 4 for QUERY TRAN command

TSO SPOC input:

```
QUERY TRAN QCNT(GT,0) SHOW(AFFIN)
```

TSO SPOC output:

```
  Trancode MbrName    CC     QCnt Affinity
  APOL12   IMS1        0        1
  APOL12   IMS1        0        1 IMS1
```

Explanation: This example queries transaction messages on the shared queue and shows which messages have affinity status.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY MODIFY command (Commands)

/DISPLAY STATUS command (Commands)

/DISPLAY TRAN command (Commands)

Command keywords and their synonyms (Commands)

QUERY TRANDESC command

Use the QUERY TRANDESC command to query information about transaction descriptors.

A descriptor is a model that can be used to create descriptors or resources.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 319
- "Usage notes" on page 324
- "Output fields" on page 324
- "Return, reason, and completion codes" on page 331
- "Examples" on page 332

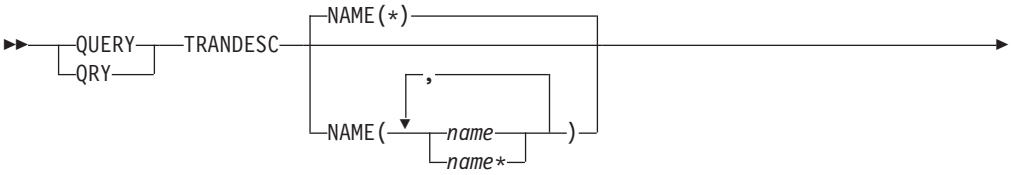
Environment

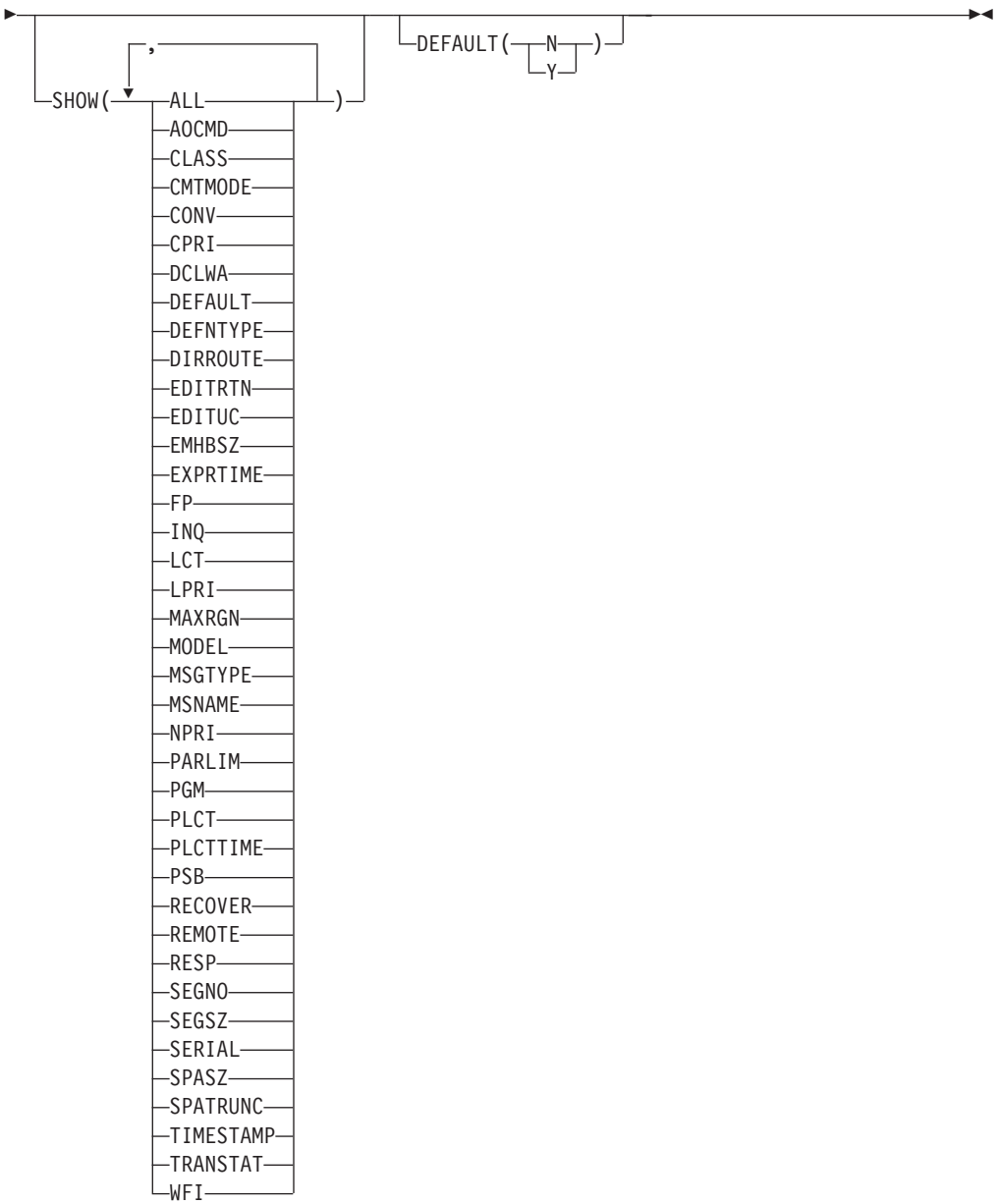
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 144. Valid environments for the QUERY TRANDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY TRANDESC	X		X
NAME	X		X
SHOW	X		X
DEFAULT	X		X

Syntax





Keywords

The following keywords are valid for the QUERY TRANDESC command:

DEFAULT
Selects transaction descriptors for display that possess the default value specified. DEFAULT(Y) displays the only default descriptor. DEFAULT(N) displays all the other descriptors that are not the default.

NAME
Specifies the 1-8 character name of the transaction descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all transaction descriptors.

SHOW
Specifies the transaction descriptor output fields to be returned. The

transaction descriptor name is always returned along with the name of the IMS that created the output and the completion code. The filters that are supported with the SHOW keyword are:

ALL
Returns all information about the transaction descriptor itself.

AOCMD
Specifies that you want the AOI option returned which indicates whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call. If AOCMD is defined as CMD, TRAN, or Y, and the AOI1 execution parameter is defined as AOI1=N, no authorization checking is done and the transaction is permitted to issue CMD and ICMD calls.

CLASS
Scheduling class used to select a transaction for scheduling. A transaction can be scheduled if there is a message processing region available for that class.

CMTMODE
Specifies when database updates and non-express output messages are committed. This operand affects emergency restart.

CONV
Conversation option indicates whether the transaction is conversational.

CPRI
The current priority. The current priority is the normal priority, when the transaction queue count is less than the limit count. The current priority is raised to the limit priority if the transaction queue count is equal to, or exceeds, the limit count. The current priority for transaction descriptors is always 0 because the current priority is a runtime scheduling value that does not apply to transaction descriptors, which are never scheduled.

DCLWA
Log write-ahead option.

DEFAULT
Default descriptor option.

N The descriptor is not the default.

Y The descriptor is the default. When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a default transaction descriptor called DFSDSTR1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor.

DEFNTYPE
Definition type. This is how the descriptor or resource was defined.

DIRROUTE
MSC directed routing option.

EDITRTN
Input edit routine that edits messages before the program receives the message.

EDITUC
The uppercase translation of input data.

EMHBSZ
EMH buffer size required to run the Fast Path transaction.

EXPRTIME
Transaction expiration time in seconds.

FP Fast Path option.

INQ
Inquiry option.

LCT

Limit count. Specifies the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction. The limit count value can range from 1 through 65 535.

The limit count value is ignored for a transaction processed by a BMP.

The limit count value is ignored in a shared-queues environment.

LPRI

Limit priority. The scheduling priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class.

The limit priority value is ignored for a transaction processed by a BMP.

MAXRGN
Maximum region count. This limits the number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction. When the number of MPP regions is not limited, one transaction might monopolize all available regions. MAXRGN(0) means that no limit is imposed.

MODEL
The model name and model type used to create this resource or descriptor. If the descriptor or resource is created with one or more attributes defined and no model specified, the model name and model type is the default descriptor. The model name and model type are blank for the IMS-defined transaction descriptor DFSDSTR1. The CREATE command specified without the LIKE keyword creates a descriptor or resource using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSTR1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor or resource using a model. The descriptor or resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor or resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

MSGTYPE

Message segment type (single or multiple segment). It specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing.

If MSC-directed routing is used in a multiple IMS system configuration, IMS does not ensure that both the message and the transaction destined to process that message are either single segment or multiple segments.

MSNAME

The logical link path name, remote system ID, and local system ID are returned.

Logical link path name in a multiple IMS system configuration (MSC). A logical link path is a path between any two IMS systems. The IMS systems are identified by the remote system ID and the local system ID associated with the logical link path. The remote system ID identifies the system in which messages using this path are to be processed. The local system ID identifies this system.

The remote system ID (SIDR) identifies the IMS system on which the application program runs. A value of 0 means MSC is not enabled on this system. The local system ID and remote system ID are the same for local transactions.

The local system ID (SIDL) identifies the originating system to which responses are returned. A value of 0 means MSC is not enabled on this system. The local system ID and remote system ID are the same for local transactions.

NPRI

Normal scheduling priority. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The normal priority is assigned to the transaction as the scheduling priority when the number of input transactions enqueued and waiting to be processed is less than the limit count value.

The normal priority value is ignored for a transaction processed by a BMP.

PARLIM

Parallel processing limit count. This is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. This is the threshold value to be used when the associated program is defined with a scheduling type of parallel. An additional region is scheduled whenever the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.

PARLIM(0) indicates that any input message can cause a new region to be scheduled. PARLIM(65535) indicates that parallel processing is disabled and IMS allows the transaction to be scheduled in only one region at a time.

PGM

The name of the program associated with this transaction. This name matches the PSB name in ACBLIB.

PLCT

Processing limit count. This is the maximum number of messages sent to the application program by the IMS for processing without reloading the application program.

- PLCT(0) means that a maximum of one message is sent to the application program at a single program load.
- PLCT(65535) means that no limit is placed on the number of messages processed at a single program load.

PLCTTIME

Processing limit count time. This is the amount of time (in hundredths of seconds) allowable to process a single transaction (or message). The number specifies the maximum CPU time allowed for each message to be processed in the message processing region.

PLCTTIME(6553500) means that no time limit is placed on the application program.

PSB

The name of the program associated with this transaction. This name matches the PSB name in ACBLIB.

RECOVER

Recovery option.

REMOTE

Remote option.

RESP

Response-mode option.

SEGNO

Segment number. This is the maximum number of application program output segments that are allowed into the message queues per Get Unique (GU) call from the application program.

SEGNO(0) means that the number of segments is not checked by the online system at execution time.

SEGSZ

Segment size. This is the maximum number of bytes allowed in any one output segment.

SEGSZ(0) means that the segment size is not checked by the online system at execution time.

SERIAL

Serial option.

SPASZ

Scratchpad area size for a conversational transaction.

SPATRUNC

The SPA data truncation option indicates whether the SPA data should be truncated or preserved across a program switch to a transaction that is defined with a smaller SPA.

When a conversation initially starts and a program switch occurs, the SPATRUNC option is checked and set or reset as specified. When the

option is set, it remains set for the life of the conversation, or until a program switch occurs to a transaction that specifies the option is to be reset.

When a program switch occurs, the truncated data option for the new transaction is first checked, and that specification is set for the conversation and is used for the SPA inserted into the output message. If the option is not specified for the new transaction, the option currently in effect for the conversation is used.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

TRANSTAT
Transaction level statistics option.

WFI
Wait-for-input option.

Usage notes

This command can be issued only through the Operations Manager API. This command applies to DB/DC and DCCTL systems. This command is allowed on XRF alternate and RSR tracker systems. The QUERY TRANDESC command is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

Output fields

The following table shows the QUERY TRANDESC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 145. Output fields for the QUERY TRANDESC command

Short label	Keyword	Scope	Meaning
AOCMD	AOCMD	LCL	Indicates whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call. **CMD** Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, authorization checking is based on which transactions can issue a particular command. In this case, the commands (or the first three characters of the commands) need to be defined to RACF or an equivalent product as a user. The type-1 AOI transactions must be defined as profiles under the TIMS class, and for each transaction, the commands it can issue must be specified. **N** Indicates that the transaction is not permitted to issue type-1 AOI CMD calls. The transaction is permitted to issue type-2 AOI ICMD calls. **TRAN** Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the transaction code is used for authorization. The first authorization check results in the accessor environment element (ACEE) being built. This environment is kept for use by future authorization checks. The type-1 AOI transaction needs to be defined to RACF or an equivalent product as a user. The transactions will then be specified on RACF PERMIT statements for each command they are allowed to issue from a type-1 AOI transaction. Specifying AOI transactions as users to RACF might conflict with the name of a user already defined to RACF. If this occurs, then either the transaction name or the existing user name needs to be changed. **Y** Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the user ID or the program name is used for authorization. For some environments, if a Get Unique call has not yet occurred, the program name is used for authorization.
CCTXT	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.

Table 145. Output fields for the QUERY TRANDESC command (continued)

Short label	Keyword	Scope	Meaning
CMTM	CMTMODE	LCL	Commit mode for the transaction: commit after a single message (SNGL) or multiple messages (MULT). **MULT** Database updates and non-express output messages are committed only when the application program terminates normally, when the processing limit count has been reached, or, in the case of a pseudo-WFI dependent region, when there are no more messages on the queue. For example, if five transactions are processed during a single schedule of a program, all five are committed only when the fifth one is completed and the program terminates. Until a transaction has been committed, locks for updated database records are not released and non-express output messages are not queued for output. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing. **SNGL** Database updates and non-express output messages are committed when the application program completes processing each transaction. IMS invokes commit processing either when the application program requests the next message (issues a GU to the IO-PCB), or when the application program terminates. If an application ends abnormally before committing its message, emergency restart requeues the message that was in process before the abend and makes it available for reprocessing.
CONV	CONV	LCL	Conversation option. Transaction is conversational (Y), or not (N). **N** Transaction is not conversational. **Y** Transaction is conversational. The transaction message is destined for a conversational program. A conversational program processes transactions made up of several steps. A conversational program receives a message from a terminal, replies to the terminal, but saves the data from the transaction in a scratchpad area (SPA). When the person at the terminal enters more data, the program has the data it saved from the last message in the SPA, so it can continue processing the request without the person at the terminal having to enter the data again.

Table 145. Output fields for the QUERY TRANDESC command (continued)

Short label	Keyword	Scope	Meaning
DCLW	DCLWA	LCL	Perform log write-ahead for recoverable, nonresponse mode input messages and transaction output messages (Y) or not (N). **N** IMS does not perform log write-ahead. **Y** IMS performs log write-ahead for recoverable, nonresponse input messages and transaction output messages. If not defined for the transaction, the default is the DCLWA parameter in the IMSCTRL macro. This ensures that a nonresponse input transaction is made recoverable across IMS failures, prior to IMS acknowledging receipt of the input. Database changes are made recoverable prior to IMS sending associated output reply messages. This ensures that information in the log buffers is written to the IMS log, before the associated input acknowledgment or output reply is sent to the terminal.
DESC	TRANDESC	LCL	Transaction descriptor name.
DFNT	DEFNTYPE	LCL	Definition type (CREATE, IMPORT, UPDATE).
DFLT	TRANDESC	LCL	Default descriptor (Y) or not (N).
DRRT	DIRROUTE	LCL	Supports MSC directed routing (Y) or not (N). **N** The application program processing a transaction is not informed of the system which originated the transaction. The name of the originating LTERM is placed in the I/O PCB. **Y** The application program processing a transaction is informed of the system which originated the transaction, if MSC directed routing is used in a multiple IMS system configuration. An MSNAME corresponding to a logical path back to the originating system is placed in the I/O PCB.
EDTR	EDITRTN	LCL	Input edit routine name.
EDTT	EDITUC	LCL	Input data is to be translated to uppercase (Y) or not (N). **N** Input data is not translated to uppercase. It can consist of uppercase and lowercase characters as entered from the terminal. **Y** Input data is to be translated to uppercase before it is presented to the processing program. If FP(Y), the transaction is to be translated to uppercase before being presented to the edit/routing exit routine. Specifying EDITUC(Y) for VTAM terminals prevents the transmission of embedded device control characters.
EMHBS	EMHBSZ	LCL	EMH buffer size.

Table 145. Output fields for the QUERY TRANDESC command (continued)

Short label	Keyword	Scope	Meaning
FP	FP	LCL	Fast Path potential candidate (P), Fast Path exclusive (E), or FP option not enabled (N). **E** Fast Path exclusive transaction. Any message for this transaction is always routed to a Fast Path application program. **N** Fast Path option is not enabled. When FP(N) is specified, any attempt to use Fast Path resources or commands will yield unpredictable results. **P** Fast Path potential transaction. Any message for this transaction can potentially be routed to a Fast Path application program.
INQ	INQ	LCL	Inquiry transaction (Y) or not (N). **N** Inquiry option is disabled. **Y** Inquiry option is enabled. This is an inquiry transaction that, when entered, does not cause a change in any database. Programs are prohibited from issuing ISRT, DLET, or REPL calls to a database when scheduled to process a transaction defined as INQ(Y). An application program cannot do an SQL INSERT, DELETE, or UPDATE when the IMS transaction is defined with INQ(Y).
LCLS	CLASS	LCL	Scheduling class used to determine which message regions can process the transaction locally on a particular IMS.
LCP	CPRI	LCL	Local current scheduling priority. The current scheduling priority is used to calculate which transaction is selected for scheduling.
LLCT	LCT	LCL	Limit count in the local IMS. The limit count is the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction.
LLP	LPRI	LCL	Local limit scheduling priority. The limit scheduling priority is the priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value.
LMRG	MAXRGN	LCL	Local maximum region count. The maximum region count is the maximum number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction that is eligible for parallel scheduling.
LNP	NPRI	LCL	Local normal scheduling priority. The normal scheduling priority is the priority assigned to this transaction when the number of input transactions enqueued and waiting to be processed is less than the limit count value.
LPLCT	PLCT	LCL	Local processing limit count. The processing limit count is the number of transaction messages a program can process in a single scheduling.

Table 145. Output fields for the QUERY TRANDESC command (continued)

Short label	Keyword	Scope	Meaning
LPLM	PARLIM	LCL	Local parallel processing limit count. The parallel limit count is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. An additional message region is scheduled whenever the transaction queue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.
LSNO	SEGNO	LCL	Local application program output segment limit allowed in message queues for each GU call.
LSSZ	SEGSZ	LCL	Local application program output segment size limit allowed in the message queues for each GU call.
MDLN	MODEL	LCL	Model name. Name of the resource or descriptor used as a model to create this descriptor. DFSDSTR1 is the IMS descriptor name for transactions.
MDLT	MODEL	LCL	Model type, either RSC or DESC. RSC means the descriptor was created using another resource as a model. DESC means the descriptor was created using a descriptor as a model.
MSGT	MSGTYPE	LCL	Message type of single segment (SNGLSEG) or multiple segment (MULTSEG). **MULTSEG** Specifies that the incoming message can be more than one segment in length. It is not eligible for scheduling to an application program until an end-of-message indication is received, or a complete message is created by MFS. **SNGLSEG** Specifies that the incoming message is one segment in length. It becomes eligible for scheduling when the terminal operator indicates end-of-segment.
MSN	MSNAME	LCL	Logical link path name.
PLCTT	PLCTTIME	LCL	Processing limit count time.
PSB	PGM or PSB	LCL	Program name associated with the transaction.
RCV	RECOVER	LCL	Recovered during an IMS emergency or normal restart (Y) or not (N). **N** Recovery option is disabled. The transaction is not recovered. **Y** Recovery option is enabled. The transaction is recovered during IMS emergency or normal restart.
RMT	REMOTE	LCL	Remote transaction (Y) or not (N). **N** Local transaction. The transaction runs on the local system. **Y** Remote transaction. The transaction runs on a remote system.

Table 145. Output fields for the QUERY TRANDESC command (continued)

Short label	Keyword	Scope	Meaning
RSP	RESP	LCL	Response mode transaction (Y) or not (N). **N** Response mode option is disabled. For terminals specifying or accepting a default of OPTIONS=TRANRESP, input should not stop after this transaction is entered. **Y** Response mode option is enabled. For terminals specifying or accepting a default of OPTIONS=TRANRESP, no additional messages are to be allowed after this transaction is entered until this transaction sends a response message back to the terminal. Response mode can be forced or negated by individual terminal definition. RESP(Y) is ignored during online processing for all terminals that do not operate in response mode.
SER	SERIAL	LCL	Transaction is processed serially (Y) or not (N). **N** Serial option is disabled. Messages for the transaction are not processed serially. Message processing can be processed in parallel. Messages are placed on the suspend queue after a U3303 pseudoabend. Scheduling continues until repeated failures result in the transaction being stopped with a USTOP. **Y** Serial option is enabled. Messages for the transaction are processed serially. U3303 pseudoabends do not cause the message to be placed on the suspend queue but rather on the front of the transaction message queue, and the transaction is stopped with a USTOP. The USTOP of the transaction is removed when the transaction or the class is started with a /START command.
SIDL	MSNAME	LCL	Local system ID.
SIDR	MSNAME	LCL	Remote system ID.
SPASZ	SPASZ	LCL	Conversational transaction scratchpad area size.
SPATR	SPATRUNC	LCL	Conversational transaction SPA data should be truncated (R) or preserved (S) across a program switch to a transaction that is defined with a smaller SPA. **S** IMS preserves all of the data in the SPA, even when a program switch is made to a transaction that is defined with a smaller SPA. The transaction with the smaller SPA does not see the truncated data, but when the transaction switches to a transaction with a larger SPA, the truncated data is used. **R** Truncated data is not preserved.
TLS	TRANSTAT	LCL	Transaction level statistics logged (Y) or not (N). **N** Transaction level statistics logging is not active. **Y** Transaction level statistics logging is active.

Table 145. Output fields for the QUERY TRANDESC command (continued)

Short label	Keyword	Scope	Meaning
TMAC	TIMESTAMP	LCL	The time that the descriptor was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. For a transaction descriptor, the following action updates the last access time: • CREATE command or DFSINSX0 exit refers to the descriptor as a model.
TMCR	TIMESTAMP	LCL	The time that the descriptor was created. This is the result of a CREATE TRANDESC command, IMPORT command that creates the transaction descriptor, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMIM	TIMESTAMP	LCL	The time that the descriptor was last imported. The import time is retained across warm start and emergency restart.
TMUP	TIMESTAMP	LCL	The time that the descriptor was last updated as a result of an UPDATE TRANDESC command that changes a definitional attribute. This includes attributes set by the SET keyword. The update time is retained across warm start, emergency restart, EXPORT and IMPORT.
WFI	WFI	LCL	Wait-for-input transaction (Y) or not (N). **N** Wait-for-input option is disabled. **Y** Wait-for-input option is enabled. A message processing or batch processing application program that processes WFI transactions is scheduled and invoked normally. If the transaction to be processed is defined as WFI, the program is allowed to remain in main storage after it has processed the available input messages. The QC status code (no more messages) is returned to the program if the processing limit count is reached; a command is entered to change the status of the schedule transaction, database, program or class; commands relating to the databases used by the transaction are entered; or IMS is terminated with a checkpoint shutdown.

Return, reason, and completion codes

Table 146. Return and reason codes for the QUERY TRANDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001010'	No descriptors were found to be returned. The descriptor names specified might be invalid, or there were no descriptors that match the filter specified.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

Table 147. Completion codes for the QUERY TRANDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for the transaction descriptor.
10	NO RESOURCES FOUND	Transaction descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.
90	INTERNAL ERROR	

Examples

The following are examples of the QUERY TRANDESC command:

Example 1 for QUERY TRANDESC command

TSO SPOC input:

```
QRY TRANDESC SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
DescName MbrName    CC PSBname  LCls  LLCT LPLCT LPLCTTime LCPRI LNPRI LLPRI LSe
CONVDESC IMS1        0 DFSSAM04    1 65535 65535   6553500     1     1     1
DFSDSTR1 IMS1        0             1 65535 65535   6553500     1     1     1
FPEDESC  IMS1        0 EMHPSB2     1     0 65535   6553500     1     1     1
FPPDESC  IMS1        0 DFSSAM04    1 65535 65535   6553500     1     1     1
MSCDESC  IMS1        0 CPGM1B1     1 65535 65535   6553500     1     1     1
(scroll to the right screen 2)
DescName MbrName  egSz LSegNo LParLim LMaxRgn EditRtn  FP EMHBSz CmtMode MsgType
CONVDESC IMS1        0      0   65535       0          N       0 SNGL    MULTSEG
DFSDSTR1 IMS1        0      0   65535       0          N       0 SNGL    MULTSEG
FPEDESC  IMS1        0      0   65535       0          E     200 SNGL    SNGLSEG
FPPDESC  IMS1        0      0   65535       0          P     256 SNGL    SNGLSEG
MSCDESC  IMS1        0      0   65535       0          N       0 SNGL    MULTSEG
(scroll to the right screen 3)
DescName MbrName  SPATrunc SPASz  SIDR  SIDL DCLWA DirRoute EditUC Inq Recover R
CONVDESC IMS1            R   128    10    10 Y     N        Y      N   Y       N
DFSDSTR1 IMS1                  0    10    10 Y     N        Y      N   Y       N
FPEDESC  IMS1                  0    10    10 Y     N        Y      N   Y       Y
FPPDESC  IMS1                  0    10    10 Y     N        Y      N   Y       Y
MSCDESC  IMS1                  0    12    32 Y     N        Y      N   Y       N
(scroll to the right screen 4)
DescName MbrName  Resp Remote Serial WFI AOCMD Conv TranStat Dflt ModelName Mode
CONVDESC IMS1     N    N      N      N   N     Y    N        N    CDEBTRNA  RSC
DFSDSTR1 IMS1     N    N      N      N   N     N    N        Y
FPEDESC  IMS1     Y    N      N      N   N     N    N        N    DFSDSTR1  DESC
FPPDESC  IMS1     Y    N      N      N   N     N    N        N    DFSDSTR1  DESC
MSCDESC  IMS1     N    Y      N      N   N     N    N        N    DFSDSTR1  DESC
(scroll to the right screen 5)
DescName MbrName  elType MSName   TimeAccess           TimeUpdate           Time
CONVDESC IMS1                                                               2006
DFSDSTR1 IMS1                    2006.286 09:30:09.22                       2006
FPEDESC  IMS1     C                                                         2006
FPPDESC  IMS1     C                                                         2006
MSCDESC  IMS1     C      LINK31B4
(scroll to the right screen 6)
DescName MbrName  TimeCreate          TimeImport          DefnType
CONVDESC IMS1     2006.286 09:26:03.53                    CREATE
DFSDSTR1 IMS1     2006.286 09:14:54.05                    IMS
```

```
FPEDESC  IMS1     2006.286 09:29:05.52                      CREATE
FPPDESC  IMS1     2006.286 09:29:32.71                      CREATE
MSCDESC  IMS1     2006.286 09:30:09.22                      CREATE
```

OM API input:

```
CMD(QUERY TRANDESC SHOW(ALL))
```

OM API output:

```
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.286 16:35:02.217717</statime>
<stotime>2006.286 16:35:02.218321</stotime>
<staseq>BF8C42767EBF5A8E</staseq>
<stoseq>BF8C42767EE5138E</stoseq>
<rqsttkn1>USRT011 10093502</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>TRANDESC        </kwd>
<input>QRY TRANDESC SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="CONVID" llbl="ConvID" scope="LCL" sort="n" key="0"
scroll="yes" len="4" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="NODE" llbl="NodeName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="USER" llbl="UserName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="LU" llbl="LUName" scope="LCL" sort="n" key="0"
 scroll="yes"len="24" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMEM" llbl="TMember" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TPIP" llbl="TPipe" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="PSB" llbl="PSBname" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="LCLS" llbl="LCls" scope="LCL" sort="n" key="0" scroll="yes"
 len="3" dtype="INT" align="right" />
<hdr slbl="LLCT" llbl="LLCT" scope="LCL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" />
<hdr slbl="LPLCT" llbl="LPLCT" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="PLCTT" llbl="LPLCTTime" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="INT" align="right" />
<hdr slbl="LCP" llbl="LCPRI" scope="LCL" sort="n" key="0" scroll="yes*
 len="2" dtype="INT" align="right " />
<hdr slbl="LNP" llbl="LNPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="LLP" llbl="LLPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="LSSZ" llbl="LSegSz" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
```

```
                        <hdr slbl="LSNO" llbl="LSegNo" scope="LCL" sort="n" key="0"
                         scroll="yes" len="5" dtype="INT" align="right" />
                        <hdr slbl="LPLM" llbl="LParLim" scope="LCL" sort="n" key="0"
                         scroll="yes" len="5" dtype="INT" align="right" />
                        <hdr slbl="LMRG" llbl="LMaxRgn" scope="LCL" sort="n" key="0"
                         scroll="yes" len="5" dtype="INT" align="right" />
                        <hdr slbl="EDTR" llbl="EditRtn" scope="LCL" sort="n" key="0"
                         scroll="yes" len="8" dtype="CHAR" align="left" />
                        <hdr slbl="FP" llbl="FP" scope="LCL" sort="n" key="0" scroll="yes"
                         len="1" dtype="CHAR" align="left" />
                        <hdr slbl="EMHBS" llbl="EMHBSz" scope="LCL" sort="n" key="0"
                         scroll="yes" len="5" dtype="INT" align="right" />
                        <hdr slbl="CMTM" llbl="CmtMode" scope="LCL" sort="n" key="0"
                         scroll="yes" len="4" dtype="CHAR" align="left" />
                        <hdr slbl="MSGT" llbl="MsgType" scope="LCL" sort="n" key="0"
                         scroll="yes" len="7" dtype="CHAR" align="left" />
                        <hdr slbl="SPATR" llbl="SPATrunc" scope="LCL" sort="n" key="0"
                        scroll="yes" len="1" dtype="CHAR" align="right" />
                        <hdr slbl="SPASZ" llbl="SPASz" scope="LCL" sort="n" key="0"
                         scroll="yes" len="5" dtype="INT" align="right" />
                        <hdr slbl="SIDR" llbl="SIDR" scope="LCL" sort="n" key="0" scroll="yes"
                         len="5" dtype="INT" align="right" />
                        <hdr slbl="SIDL" llbl="SIDL" scope="LCL" sort="n" key="0" scroll="yes"
                         len="5" dtype="INT" align="right" />
                        <hdr slbl="DCLW" llbl="DCLWA" scope="LCL" sort="n" key="0" scroll="yes"
                         len="1" dtype="CHAR" align="left" />
                        <hdr slbl="DRRT" llbl="DirRoute" scope="LCL" sort="n" key="0"
                         scroll="yes" len="1" dtype="CHAR" align="left" />
                        <hdr slbl="EDTT" llbl="EditUC" scope="LCL" sort="n" key="0"
                         scroll="yes" len="1" dtype="CHAR" align="left" />
                        <hdr slbl="INQ" llbl="Inq" scope="LCL" sort="n" key="0" scroll="yes"
                         len="1" dtype="CHAR" align="left" />
                        <hdr slbl="RCV" llbl="Recover" scope="LCL" sort="n" key="0"
                         scroll="yes" len="1" dtype="CHAR" align="left" />
                        <hdr slbl="RSP" llbl="Resp" scope="LCL" sort="n" key="0" scroll="yes"
                         len="1" dtype="CHAR" align="left" />
                        <hdr slbl="RMT" llbl="Remote" scope="LCL" sort="n" key="0" scroll="yes"
                         len="1" dtype="CHAR" align="left" />
                        <hdr slbl="SER" llbl="Serial" scope="LCL" sort="n" key="0" scroll="yes"
                         len="1" dtype="CHAR" align="left" />
                        <hdr slbl="WFI" llbl="WFI" scope="LCL" sort="n" key="0" scroll="yes"
                         len="1" dtype="CHAR" align="left" />
                        <hdr slbl="AOCMD" llbl="AOCMD" scope="LCL" sort="n" key="0"
                         scroll="yes" len="4" dtype="CHAR" align="left" />
                        <hdr slbl="CONV" llbl="Conv" scope="LCL" sort="n" key="0" scroll="yes"
                         len="1" dtype="CHAR" align="left" />
                        <hdr slbl="TLS" llbl="TranStat" scope="LCL" sort="n" key="0" scroll="y
                         es" len="1" dtype="CHAR" align="left" />
                        <hdr slbl="DFLT" llbl="Dflt" scope="LCL" sort="n" key="0" scroll="yes"
                         len="1" dtype="CHAR" align="left" />
                        <hdr slbl="MDLN" llbl="ModelName" scope="LCL" sort="n" key="0"
                         scroll="yes" len="8" dtype="CHAR" align="left" />
                        <hdr slbl="MDLT" llbl="ModelType" scope="LCL" sort="n" key="0"
                         scroll="yes" len="8" dtype="CHAR" align="left" />
                        <hdr slbl="MSN" llbl="MSName" scope="LCL" sort="n" key="0" scroll="yes"
                         len="8" dtype="CHAR" align="left" />
                        <hdr slbl="TMAC" llbl="TimeAccess" scope="LCL" sort="n" key="0"
                         scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
                        <hdr slbl="TMUP" llbl="TimeUpdate" scope="LCL" sort="n" key="0"
                         scroll="yes" len="20" dtype="CHAR" align="left" />
                        <hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
                         scroll="yes" len="20" dtype="CHAR" align="left" />
                        <hdr slbl="TMIM" llbl="TimeImport" scope="LCL" sort="n" key="0"
                        scroll="yes" len="20" dtype="CHAR" align="left" />
                        <hdr slbl="DFNT" llbl="DefnType" scope="LCL" sort="n" key="0"
                         scroll="yes" len="8" dtype="CHAR" align="left" />
                        </cmdrsphdr>
```

```
<cmdrspdata>
<rsp>DESC(DFSDSTR1) MBR(IMS1    ) CC(   0)  PSB(        ) LCLS(  1)
 LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
 0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N) DCLW(Y)
 DFNT(IMS) DFLT(Y) DRRT(N) EDTT(Y) EMHBS(    0) FP(N) INQ(N)
 MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(N) SER(N) SIDL( 10)
 SIDR( 10) SPASZ(    0) TMAC(2006.286 09:30:09.22) TMCR(2006.286
 09:14:54.05) TLS(N) WFI(N)</rsp>
<rsp>DESC(FPPDESC ) MBR(IMS1    ) CC(   0)  PSB(DFSSAM04) LCLS(  1)
 LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
 0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N) DCLW(Y)
 DFNT(CREATE) DFLT(N) DRRT(N) EDTT(Y) EMHBS(  256) FP(P) INQ(N)
 MDLT(DESC) MDLN(DFSDSTR1) MSGT(SNGLSEG) PLCTT(6553500) RCV(Y) RMT(N)
 RSP(Y) SER(N) SIDL( 10) SIDR( 10) SPASZ(    0) TMCR(2006.286
 09:29:32.71) TLS(N) WFI(N)</rsp>
<rsp>DESC(CONVDESC) MBR(IMS1    ) CC(   0)  PSB(DFSSAM04) LCLS(  1)
 LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
 0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(Y) DCLW(Y)
 DFNT(CREATE) DFLT(N) DRRT(N) EDTT(Y) EMHBS(    0) FP(N) INQ(N)
MDLT(RSC) MDLN(CDEBTRNA) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N)
 RSP(N) SER(N) SIDL( 10) SIDR( 10) SPASZ(  128) SPATR(R) TMCR(2006.286
 09:26:03.53) TLS(N) WFI(N)</rsp>
<rsp>DESC(MSCDESC ) MBR(IMS1    ) CC(   0)  PSB(CPGM1B1 ) LCLS(  1)
 LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
 0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N) DCLW(Y)
 DFNT(CREATE) DFLT(N) DRRT(N) EDTT(Y) EMHBS(    0) FP(N) INQ(N)
 MDLT(DESC) MDLN(DFSDSTR1) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(Y)
 RSP(N) SER(N) SIDL( 32) SIDR( 12) MSN(LINK31B4) SPASZ(    0)
 TMCR(2006.286 09:30:09.22) TLS(N) WFI(N)</rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1    ) CC(   0)  PSB(EMHPSB2 ) LCLS(  1)
 LLCT(    0) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
 0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N) DCLW(Y)
 DFNT(CREATE) DFLT(N) DRRT(N) EDTT(Y) EMHBS(  200) FP(E) INQ(N)
 MDLT(DESC) MDLN(DFSDSTR1) MSGT(SNGLSEG) PLCTT(6553500) RCV(Y) RMT(N)
 RSP(Y) SER(N) SIDL( 10) SIDR( 10) SPASZ(    0) TMCR(2006.286
 09:29:05.52) TLS(N) WFI(N)</rsp>
</cmdrspdata>
</imsout>
```

Explanation: All transaction descriptors are returned with all output fields. All the transaction descriptor output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The transaction descriptor name and the member name that built the line of output are displayed on every screen. The fields that are blank are not applicable to the specified transaction descriptor. IMS defines descriptor DFSDSTR1 at IMS cold start time to contain the IMS default values for transactions. A few other descriptors were created dynamically with CREATE TRANDESC commands, including CONVDESC for conversational transactions, FPEDESC for FP exclusive transactions, FPPDESC for FP potential transactions, and MSCDESC for remote MSC transactions. DFSDSTR1 shows the last access time for TimeAccess, which was updated by the last CREATE TRAN or CREATE TRANDESC command that referred to it as the default model.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY USER command

Use the QUERY USER command to display information about VTAM users across the IMSplex. A user is either a dynamic (ETO) user, or a static or dynamic (ETO) ISC subpool user. In this context, a user ID that is used for user or terminal security is not considered as a user, but rather as a user ID. This command can be specified only through the OM API and is valid on an XRF alternate.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 337
- "Usage notes" on page 341
- "Similar IMS type-1 commands" on page 342
- "Output fields" on page 342
- "QUERY USER status" on page 347
- "Return, reason, and completion codes" on page 348
- "Examples" on page 350

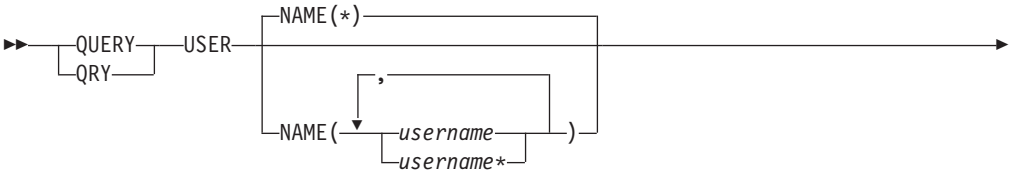
Environment

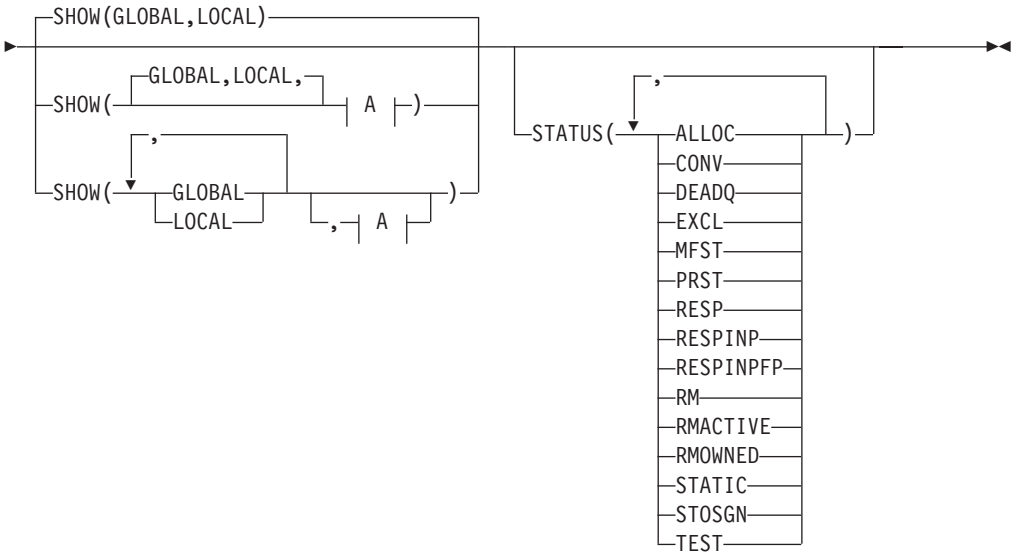
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 148. Valid environments for the QUERY USER command and keywords

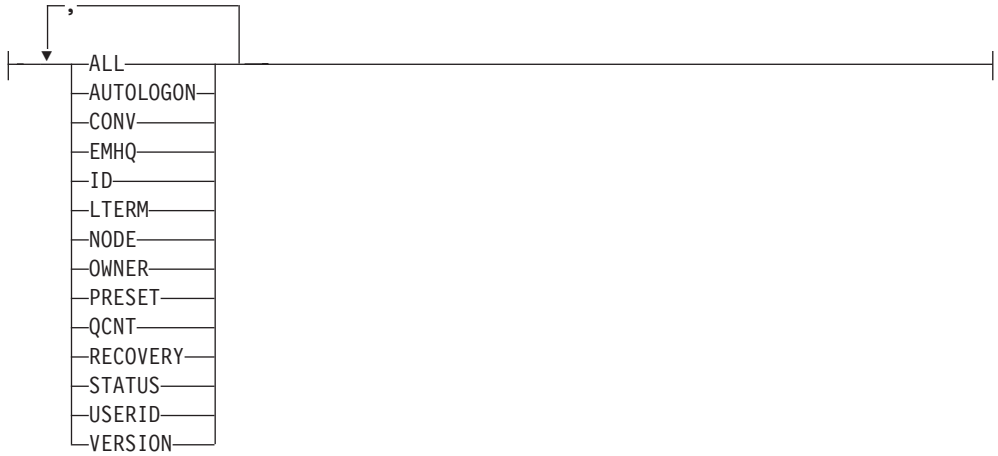
Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY USER	X		X
NAME	X		X
SHOW	X		X
STATUS	X		X

Syntax





A:



Keywords

The following keywords are valid for the QUERY USER command:

NAME()
Specifies the names of one or more users (dynamic user or ISC subpool user) that are to be displayed. Valid names are 1-8 characters, and wildcards can be specified. To display all users, specify NAME(*). NAME(*) is the default.

SHOW()
Specifies the user output fields to be returned. The user name is always returned along with the name of the IMS that created the output and the completion code. If SHOW is not specified, only the user names are returned, provided that the STATUS filter is not specified. This provides a method for a system management application to obtain a list of all user names that are currently known in the IMSplex.

Two parameters, GLOBAL and LOCAL, are used to specify the location (global resources or local resources) where IMS should obtain the information that is

to be displayed. The default is both GLOBAL and LOCAL. The rest of the parameters are used to specify what information is displayed.

The parameters supported with the SHOW keyword, which can be specified in any order, are:

ALL
Returns all of the output fields, except for those fields displayed when the LTERM and CONV parameters are specified. To display lterm and conversation information with all other output, specify SHOW(ALL,LTERM,CONV).

AUTOLOGON
Displays the current autologon information for the specified user. This includes the name of the node with which to establish a session, the VTAM mode table name, the name of the node descriptor to be used to build the node, and (if ISC) the identifier of the other system half-session qualifier.

CONV
The conversation ID, transaction, and conversation status associated with the user. Each conversation is returned on a separate command response display line. Conversation status is not included when SHOW(ALL) is specified. To display conversation information with all other output, specify SHOW(ALL,CONV).

EMHQ
Displays the user message queue count in the Expedited Message Handler (EMH) queues. The queue count is the sum of the queue counts for each lterm associated with the specified user.

EMHQ is valid only when the GLOBAL parameter is specified on the SHOW keyword. If GLOBAL is not specified, then the EMHQ parameter is ignored.

EMHQ is processed by the command master only. It is ignored by all other IMS systems.

EMHQ is valid only if shared EMH is used in a shared queues environment. Otherwise, this parameter is ignored.

If the user resource exists in the resource structure, then the lterms associated with the global user resource are used to obtain the queue counts from the EMH queues. Otherwise, the lterms associated with the local user resource are used, but if the user does not exist locally on the command master, then the queue count is 0.

GLOBAL
When GLOBAL is specified, the command master displays global information, depending on the other SHOW parameters specified. This includes information from shared queues and the resource structure.

The GLOBAL parameter is processed by the command master only. All other IMS systems ignore this parameter. If LOCAL is not also specified, then all IMS systems other than the command master ignore the command.

GLOBAL is applicable only when the command master is using shared queues or sysplex terminal management (or both). GLOBAL is not applicable when the command master is not using shared queues or sysplex terminal management. In this environment, if LOCAL is also specified, then GLOBAL is ignored. Otherwise, the command master rejects the command.

If shared queues are enabled, and global queue counts are requested, then the command master will make requests to CQS to determine the appropriate queue counts. This includes both MSGQ and EMHQ.

If sysplex terminal management is enabled, then the command master will make requests to RM to determine the appropriate global status.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

ID Displays the other half-session qualifier name of the ISC node.

LOCAL
When LOCAL is specified, then all IMS systems including the command master display local information, depending on the other SHOW parameters specified. This includes information local to the IMS processing the command.

The LOCAL parameter is processed by all IMS systems, including the command master.

LOCAL is applicable in any environment, regardless of whether shared queues or sysplex terminal management are used.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

LTERM
Displays the logical terminal (lterm) names, if any, associated with the user. A user might have zero or more logical terminals associated with it. Each lterm associated with the user is returned on a separate command response line.

The lterm status is not included when SHOW(ALL) is specified. To display lterm information with all other output, specify SHOW(ALL,LTERM).

NODE
Displays the dynamic or ISC node that the user is associated with.

OWNER
Displays the owner of the user resource in the resource structure. This applies only when sysplex terminal management is enabled, and is processed by the command master only. All other IMS systems ignore this parameter.

The owner is the IMSID (or RSENAME for XRF systems) of the IMS system that owns the user. An IMS system owns a user resource if the resource is active (the user is signed on), or an IMS system is maintaining significant status for that resource.

PRESET
Displays the preset destination name for a user. A user is in preset destination mode following a /SET command. The preset destination name is either a transaction name or a logical terminal (LTERM) name. All messages entered from this user are sent to the preset destination transaction or LTERM.

QCNT
Displays the user message queue count. The queue count is the sum of the queue counts for each lterm associated with the specified user.

The local queue counts value returned on this command represents the messages being processed by the IMS system where this command is issued. In a shared-queues environment, do not use the local queue counts for queue evaluation purposes. The global queue counts, which are returned on this command, represent the current message queue counts on the shared queues structure.

When the LOCAL parameter is also specified on the SHOW keyword, then all IMS systems that process the command, including the command master, display the local queue count. This is valid whether shared message queues are enabled.

When the GLOBAL parameter is also specified on the SHOW keyword, and shared message queues are enabled, then the command master displays the global queue count on the shared message queues (MSGQ). If the user resource exists in the resource structure, then the lterms associated with the global user resource are used to obtain the queue counts from shared queues. Otherwise, the lterms associated with the local user resource are used, but if the user does not exist locally on the command master, then the queue count is 0.

The local and global queue counts are displayed as separate output fields.

RECOVERY
Displays the status recovery mode (SRM) and level of recovery for the user. End-user significant status can be conversation, Fast Path, full-function response mode, or STSN (set-and-test-sequence-number) status.

SRM determines where end-user significant status, if any exists, is recovered following a user signoff, or session or IMS termination. The output displays the SRM for the user as either GLOBAL (sysplex terminal management recovers it in the resource structure), LOCAL (IMS recovers it locally, which indicates an affinity to a particular IMS), or NONE (status is discarded).

Level of recovery determines what end-user significant status, if any exists, is recovered (if SRM is GLOBAL or LOCAL). The output displays whether conversation status is recovered (RCVYCONV), Fast Path status is recovered (RCVYFP), full-function response mode is recovered (RCVYRESP), and STSN status is recovered (RCVYSTSN).

STATUS
Returns local or global status of the user. See “QUERY USER status” on page 347 for a list and meaning of possible status that might be returned.

USERID
The RACF user ID that was used to sign the user on to a node.

VERSION
Displays the RM version number of the user resource. This is the version number assigned to the user, which is assigned by MVS, and maintained by RM, when the resource is created or updated in the resource structure. VERSION applies only when sysplex terminal management is enabled. VERSION is ignored when sysplex terminal management is not enabled.

STATUS()

Selects users for display that possess at least one of the specified user statuses. The status might exist locally or globally if sysplex terminal management (STM) is enabled.

The STATUS filter is valid in both a sysplex terminal management environment and in a non-sysplex terminal management environment. In a sysplex terminal management environment, the status selected might exist locally, globally, or both. If sysplex terminal management is not enabled, then the status only exists locally.

If SHOW(LOCAL) is specified, then IMS will select only those users with the appropriate status in the local system. The command is processed by all IMS systems, including the command master.

If SHOW(GLOBAL) is specified, and sysplex terminal management is enabled, then IMS will select only those users with the appropriate status in the resource structure. The command is processed only by the command master. If SHOW(GLOBAL) is specified, but sysplex terminal management is not enabled, then the command is rejected.

If SHOW(GLOBAL,LOCAL) is specified (the default), then IMS will select those users with the appropriate status either locally or in the resource structure (if sysplex terminal management is enabled). The command is processed by all IMS systems. The command master processes both global and local information.

The output returned when the status filter is specified includes the status of the user, even if SHOW(STATUS) is not specified.

To determine which filters can be used to select users with corresponding status, see "QUERY USER status" on page 347.

Usage notes

The QUERY USER command can be specified only through the OM API.

The QUERY USER command can be issued on an XRF alternate system, but SHOW(GLOBAL) is not supported. Only local information can be displayed.

The processing of the QUERY USER command is different depending on whether IMS sysplex terminal management is enabled.

- If IMS sysplex terminal management is not enabled, processing is local for each system. The results of type-1 and type-2 commands are similar.
- If IMS sysplex terminal management is enabled, type-1 and type-2 command processing is similar when displaying local information. However, they differ in how global information is displayed.
- For type-1 /DISPLAY commands with IMS sysplex terminal management enabled, the command master displays information from either the resource structure or the local system, but not both. If the resource being displayed is not owned by any system or is owned by the command master, the command master displays the global resource. However, if the resource is owned by a system other than the command master, the command master displays only the local resource, and the owning system is responsible for displaying the global resource.
- For type-2 QUERY commands with IMS sysplex terminal management enabled, the command master is the only system that displays global resource information, regardless of whether the resource is owned. In addition, the

command master displays local resource information. All other IMS systems that process the command display local resource information only. This approach allows more flexibility in displaying all information in an IMSplex.

The SHOW keyword determines which IMS systems process the command, and what information is displayed.

- If SHOW(GLOBAL) is specified, then the command master displays global information, which includes the global queue count if shared queues are enabled, and status from the resource structure if sysplex terminal management is enabled (STM=YES defined in DFSDCxxx PROCLIB member). This is true whether or not the user is active on any particular IMS system. All other IMS systems to which OM routes the command ignore the GLOBAL parameter with return code X'00000004' and reason code X'00001000'.
- If SHOW(LOCAL) is specified, then each IMS system to which OM routes the command (including the command master) processes the command, and displays information that is local to each system.
- If both GLOBAL and LOCAL are specified (which is the default), then the command master displays both global and local information, and all other IMS systems to which OM routes the command displays local information.

Similar IMS type-1 commands

The following table shows variations of the QUERY USER command and the type-1 IMS commands that perform similar functions.

Table 149. Type-1 equivalents for the QUERY USER command

QUERY USER command	Similar IMS type-1 command
QUERY USER SHOW(AUTOLOGON)	/DISPLAY USER user AUTOLOGON
QUERY USER SHOW(CONV)	/DISPLAY CONV USER user
QUERY USER SHOW(EMHQ)	/DISPLAY USER user QCNT EMHQ
QUERY USER SHOW(ID)	/DISPLAY ASMT USER user
QUERY USER SHOW(LTERM)	/DISPLAY ASMT USER user
QUERY USER SHOW(NODE)	/DISPLAY ASMT USER user /DISPLAY USER user
QUERY USER SHOW(OWNER)	/DISPLAY USER user RECOVERY
QUERY USER SHOW(PRESET)	/DISPLAY USER user
QUERY USER SHOW(QCNT)	/DISPLAY USER user /DISPLAY USER user QCNT
QUERY USER SHOW(RECOVERY)	/DISPLAY USER user RECOVERY
QUERY USER SHOW(STATUS)	/DISPLAY USER user
QUERY USER SHOW(USERID)	/DISPLAY ASMT USER user /DISPLAY USER user
QUERY USER STATUS(CONV)	/DISPLAY CONV
QUERY USER STATUS(status)	/DISPLAY STATUS USER

Output fields

The following table shows the QUERY USER output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

SHOW parameter
Identifies the parameter on the SHOW keyword that caused the field to be generated. *Error* appears for output fields that are returned for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field. GBL indicates that the field can be generated only by the command master when displaying global information for SHOW(GLOBAL). LCL indicates that the field can be generated by any IMS displaying local information for SHOW(LOCAL). N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 150. Output fields for the QUERY USER command

Short label	Long label	SHOW parameter	Scope	Meaning
AUTID	AutLID	AUTOLOGON	GBL	Identifier of the other system half-session qualifier for autologon in the resource structure. This is only applicable if the user is associated with a parallel session ISC node.
AUTLD	AutLDsc	AUTOLOGON	GBL	The logon descriptor for autologon in the resource structure.
AUTMD	AutLMdt	AUTOLOGON	GBL	The VTAM mode table name for autologon in the resource structure.
AUTND	AutLNode	AUTOLOGON	GBL	The node name for autologon in the resource structure.
CC	CC	N/A	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 348 for more information. The completion code is always returned.
CCTXT	CCText	*Error*	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
CONVID	ConvID	CONV	GBL	The conversation ID for a conversation associated with the user, as it exists in the resource structure. A user might have zero, one, or more conversations. Each conversation will have its own line of output.
CONVSTT	ConvStat	CONV	GBL	The status of a conversation associated with the user, as it exists in the resource structure. A user might have zero, one, or more conversations. The status can be: • CONVHELD: Conversation is held • CONVACTV: Conversation is active • CONVSCHD: Conversation is scheduled

Table 150. Output fields for the QUERY USER command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
CONVTRN	ConvTran	CONV	GBL	The transaction for a conversation associated with the user, as it exists in the resource structure. A user might have zero, one, or more conversations.
EMHQ	EMHQCnt	EMHQ	GBL	Global lterm queue count in the EMH (Expedited Message Handler) queues. EMHQ is displayed only if shared EMH is used.
GBL	Gbl	GLOBAL	GBL	If 'Y', then the output reflects the status found globally in RM. If blank, then the output reflects the status found locally.
ID	ID	ID	GBL	For ISC parallel-session terminals, displays the global half-session qualifier of the other system.
LAUTID	LAutLID	AUTOLOGON	LCL	Identifier of the other system half-session qualifier for autologon in the local system. This is applicable only if the user is associated with a parallel session ISC node.
LAUTLD	LAutLDsc	AUTOLOGON	LCL	The logon descriptor for autologon in the local system.
LAUTMD	LAutLMdt	AUTOLOGON	LCL	The VTAM mode table name for autologon in the local system.
LAUTND	LAutLNode	AUTOLOGON	LCL	The node name for autologon in the local system.
LCONVID	LConvID	CONV	LCL	The conversation ID for a conversation associated with the user, as it exists in the local system. A user might have zero, one, or more conversations. Each conversation will have its own line of output.
LCONVSTT	LConvStat	CONV	LCL	The status of a conversation associated with the user, as it exists in the local system. A user might have zero, one, or more conversations. The status can be: • CONVHELD: Conversation is held • CONVACTV: Conversation is active • CONVSCHD: Conversation is scheduled
LCONVTRN	LConvTran	CONV	LCL	The transaction for a conversation associated with the user, as it exists in the local system. A user might have zero, one, or more conversations.
LID	LID	ID	LCL	For ISC parallel-session terminals, displays the local half-session qualifier of the other system.
LLTERM	LLterm	LTERM	LCL	Local logical terminal names. The logical terminal names associated with the user.
LNODE	LNode	NODE	LCL	Identifies the dynamic or static node associated with the user on the local system.
LPRST	LPreset	PRESET	LCL	Identifies the preset destination transaction or LTERM name when the user is in preset destination mode, which is established by the /SET command. All messages entered at this terminal are sent to the preset destination transaction code or LTERM.

Table 150. Output fields for the QUERY USER command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LQ	LQCnt	QCNT	LCL	Local queue count.
LRCVY	LRcvy	RECOVERY	LCL	The level of recovery for end-user significant status in the local system, which indicates what type of status is recoverable. Any value presented here implies that the corresponding status is recoverable. If SRM is LOCAL, the status will be recovered locally. If SRM is GLOBAL, the status will be recovered globally. These values are not applicable if SRM is NONE or there is no SRM. The status values that can be returned (more than one are possible) are: • CONV: IMS conversations are recoverable (RCVYCONV=YES). • FP: Fast Path status is recoverable (RCVYFP=YES). • RESP: Full-function response mode status is recoverable (RCVYRESP=YES). • STSN: STSN status is recoverable (RCVYSTSN=YES).
LSRM	LSRM	RECOVERY	LCL	The status recovery mode in the local system, which determines where the end-user significant status is maintained and recovered from. The output will be one of the following: • GBL: Status is saved globally in the IMS resource structure. • LCL: Status is saved in local control blocks and log records. • NONE: Status is not saved in the IMS resource structure or log records. • Blank: SRM is not yet established, or the user is not signed on and there is no end-user significant status.
LSTT	LclStat	STATUS	LCL	Local user status. See “QUERY USER status” on page 347 for a list and explanation of the possible user status.
LTERM	Lterm	LTERM	GBL	Global logical terminal names. The logical terminal names associated with the user.
LUID	LUserid	USERID	LCL	Identifies the local user ID signed on to the user.
LVER	LVersion#	VERSION	LCL	Version number for the user resource being maintained in the local system. This field applies only when STM is enabled.
MBR	MbrName	N/A	N/A	IMSplex member (modular unit) that built the output line. IMS identifier of the IMS that built the output. The IMS identifier is always returned.
NODE	Node	NODE	GBL	Identifies the dynamic or static node associated with the user in the resource structure.

Table 150. Output fields for the QUERY USER command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
OWNER	Owner	OWNER	GBL	Resource owner. IMS identifier or RSENAME of IMS where the user is active. If no owning IMS system exists and RM contains an entry for the resource, the owner field will be blank.
QCNT	QCnt	QCNT	GBL	Global queue count on the shared queues. Global queue count can only be displayed if shared queues are used.
RCVY	Rcvy	RECOVERY	GBL	The level of recovery for end-user significant status in the resource structure, which indicates what type of status is recoverable. Any value presented here implies that the corresponding status is recoverable. If SRM is LOCAL, the status will be recovered locally. If SRM is GLOBAL, the status will be recovered globally. These values are not applicable if SRM is NONE or there is no SRM. The status values that can be returned (more than one are possible) are: • CONV: IMS conversations are recoverable (RCVYCONV=YES). • FP: Fast Path status is recoverable (RCVYFP=YES). • STSN: STSN status is recoverable (RCVYSTSN=YES).
SRM	SRM	RECOVERY	GBL	The status recovery mode in the resource structure, which determines where the end-user significant status is maintained and recovered from. The output will be one of the following: • GBL: Status is saved globally in the IMS resource structure. • LCL: Status is saved in local control blocks and log records. • NONE: Status is not saved in the IMS resource structure or log records. • Blank: SRM is not yet established, or the user is not signed on and there is no end-user significant status.
STT	Status	STATUS	GBL	Global user status. See “QUERY USER status” on page 347 for a list and explanation of the possible node status.
UID	Userid	USERID	GBL	Identifies the RACF user ID signed on to the user.
USER	User	N/A	N/A	The user name. The user name is always returned.
VER	Version#	VERSION	GBL	Version number for the user resource being maintained in the resource structure. This field applies only when STM is enabled.

QUERY USER status

The following table shows the possible user status that can be displayed. The columns in the table are:

Status The user status that is displayed.

STATUS parameter
The STATUS() filter that will select users with the specified status.

Scope The scope of the status. GBL indicates that the status can be global (it exists in the resource structure when STM is enabled), and is returned with the STT short label. LCL indicates that the status can be local, and is returned with the LSTT short label.

Meaning
Provides a brief description of the status.

Table 151. QUERY USER status

Status	STATUS parameter	Scope	Meaning
ALLOC	ALLOC	GBL and LCL	A user or ISC subpool is allocated to a node.
CONVACT	CONV	GBL and LCL	An active conversation exists.
CONVHELD	CONV	GBL and LCL	All conversations are held.
DEADQ	DEADQ	LCL	The user has dead letter queues, or whose last access time was outside the limit set by the DLQT JCL parameter. The DEADQ status can be removed by signing on the user or entering the /DEQUEUE or /ASSIGN command.
EXCL	EXCL	GBL and LCL	The user is in exclusive mode set by the /EXCLUSIVE command. The exclusive mode restricts the output received by the terminal affected.
MFST	MFST	GBL and LCL	The user is in MFSTEST mode, set by the UPDATE USER START(MFST) command or the /TEST MFS command. Terminals supported by Message Format Service use format blocks from a special test library if the requested format block is in the test library; otherwise the blocks are obtained from the production library.
PRST	PRST	LCL	The user is in preset destination mode. PRST mode is established by the /SET command. All messages entered at this terminal are sent to the preset destination transaction code or logical terminal.
RESP	RESP	LCL	The user is in response mode and the response reply message is available for output or in the process of being sent.
RESPINP	RESPINP	GBL and LCL	The user is in response mode and the response mode input is still in-doubt; for example, the response reply message is not available for output.

Table 151. QUERY USER status (continued)

Status	STATUS parameter	Scope	Meaning
RESPINPFP	RESPINPFP	GBL and LCL	The user is in Fast-Path response mode and the response mode input is still in-doubt; for example, the response reply message is not available for output.
RM	RM	GBL	The user exists in the resource structure managed by RM.
RMACTIVE	RMACTIVE	GBL	The user is active (signed on) in the IMSplex, as indicated in the RM structure (RM active).
RMOWNED	RMOWNED	GBL	The user is owned by an IMS system in the IMSplex, as indicated in the RM structure (RM owned).
STATIC	STATIC	GBL and LCL	The user was defined during system definition.
STOSGN	STOSGN	GBL and LCL	The user was stopped from signing on with the UPDATE USER command or the /STOP USER command.
TEST	TEST	LCL	The user is in test mode set by the /TEST command. Test or echo mode sends any input from the terminal back to the terminal.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY USER command. The OM return and reason codes that may be returned as a result of the QUERY USER command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the QUERY USER command.

Table 152. Return and reason codes for the QUERY USER command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001000'	The command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000008'	X'00002014'	An invalid character was specified in the resource name.
X'00000008'	X'00002040'	An invalid parameter value was specified. An invalid SHOW or STATUS value might have been specified.
X'0000000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See Table 153 on page 349 for details.
X'0000000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code. See Table 153 on page 349 for details.
X'00000010'	X'00004004'	Command processing terminated because CQS was not active.

Table 152. Return and reason codes for the QUERY USER command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004005'	Command processing terminated because CQS was not connected to the queue structure.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	Command processing terminated because the resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'00004104'	Command processing terminated because RM is not available.
X'00000010'	X'00004108'	Command processing terminated because SCI is not available.
X'00000014'	X'00005004'	A DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	Command processing terminated because of an RM error.
X'00000014'	X'00005104'	Command processing terminated because of a CQS error.
X'00000014'	X'00005108'	Command processing terminated because of an SCI error.
X'00000014'	X'00005FFF'	Command processing terminated because of an internal IMS error.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 153. Completion codes for the QUERY USER command

Completion code	Completion code text	Meaning
0		The QUERY USER command completed successfully for the resource.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
98	CQS REQUEST ERROR	Global queue counts could not be obtained because of a CQS error.

Table 153. Completion codes for the QUERY USER command (continued)

Completion code	Completion code text	Meaning
1A2	User Resource is in error	The user resource was found in the resource structure, and an associated resource was needed, but it was either not found or appeared to be in error. This is normally an error condition. However, it could be a temporary condition caused by terminal or command activity. The command should be retried.

Examples

The following are examples of the QUERY USER command:

Example 1 for QUERY USER command

TSO SPOC input:

```
QRY USER NAME(USER2*,XYZ) SHOW(LOCAL)
```

TSO SPOC output:

```
User     MbrName CC CCText
USER23   IMS1     0
USER23   IMS2     0
USER24A  IMS2     0
USER24B  IMS2     0
USER24C  IMS2     0
XYZ      IMS1    10 NO RESOURCES FOUND
XYZ      IMS2    10 NO RESOURCES FOUND
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM and shared queues are irrelevant because only LOCAL information is requested. IMS1, the command master, displays only local information because no global information is requested. IMS2 displays local information only.

- USER23 exists on IMS1 and IMS2.
- USER24A exists on IMS2.
- USER24B exists on IMS2.
- USER24C exists on IMS2.
- XYZ does not exist on any system.

Example 2 for QUERY USER command

TSO SPOC input:

```
QRY USER NAME(USER2*)
```

TSO SPOC output:

```
User     MbrName CC  Gbl
USER23   IMS1     0  Y
USER23   IMS1     0
USER23   IMS2     0
USER24A  IMS1     0  Y
USER24A  IMS2     0
USER24B  IMS1     0  Y
```

```
USER24B  IMS2     0
USER24C  IMS2     0
USER25   IMS1     0  Y
USER26   IMS1     0  Y
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information only.

- USER23 exists on IMS1, IMS2 and in the resource structure.
- USER24A exists on IMS2 and in the resource structure.
- USER24B exists on IMS2 and in the resource structure.
- USER24C exists on IMS2 only.
- USER25 exists in the resource structure only.
- USER26 exists in the resource structure only.

Example 3 for QUERY USER command

TSO SPOC input:

```
QRY USER NAME(USER2*) STATUS(STATIC) SHOW(LOCAL)
```

TSO SPOC output:

```
User     MbrName CC CCText              LclStat
USER24A  IMS2     0                     ALLOC,STATIC
USER24B  IMS2     0                     ALLOC,STATIC
USER24C  IMS2     0                     STATIC
USER2*   IMS1    10 NO RESOURCES FOUND
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM and shared queues are irrelevant because no global information is requested. IMS1, the command master, displays local information only because SHOW(LOCAL) is specified. IMS2 displays local information only. All static users are displayed, and status is displayed because the STATUS filter was specified. IMS1 did not find any static users that matched the name specified.

- USER24A exists on IMS2, and is allocated to a node.
- USER24B exists on IMS2, and is allocated to a node.
- USER24C exists on IMS2, but is not allocated.

Example 4 for QUERY USER command

TSO SPOC input:

```
QRY USER NAME(USER25) SHOW(GLOBAL,CONV,LTERM,STATUS)
```

TSO SPOC output:

```
(screen 1)
User     MbrName CC  Gbl  Lterm    ConvID ConvTran ConvStat
USER25   IMS1     0  Y
USER25   IMS1     0  Y    LTERM25A
USER25   IMS1     0  Y    LTERM25B
USER25   IMS1     0  Y                 1 TRAN1A   CONVHELD
USER25   IMS1     0  Y                 2 TRAN1A   CONVHELD
USER25   IMS1     0  Y                 3 TRAN1A   CONVACTV

(scrolled right to screen 2)
User     MbrName Gbl  Status
USER25   IMS1    Y    CONVACT,RM
USER25   IMS1    Y
```

```
USER25   IMS1    Y
USER25   IMS1    Y
USER25   IMS1    Y
USER25   IMS1    Y
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global information only. IMS2 ignores the command (RC=4, RSN=x1000) because only global information is requested.

USER25 exists in the resource structure. IMS1 displays a global line that shows that the user is not active (not signed on) in the IMSplex, and has a conversation active. There are two lterms assigned to the user, and are displayed on separate output lines. There are three conversations associated with the user, and are displayed on separate output lines.

Example 5 for QUERY USER command

TSO SPOC input:

```
QRY USER NAME(USER23) SHOW(CONV,STATUS,OWNER,RECOVERY)
```

TSO SPOC output:

```
(screen 1)
User     MbrName CC Gbl  Owner  SRM  Rcvy
USER23   IMS1     0 Y    IMS2   LCL  CONV,FP
USER23   IMS1     0
USER23   IMS2     0
USER23   IMS2     0
USER23   IMS2     0

(scrolled right to screen 2)
User     MbrName Gbl  ConvID ConvTran ConvStat Status
USER23   IMS1    Y                             ALLOC,RM,RMACTIVE,RMOWNED
USER23   IMS1
USER23   IMS2
USER23   IMS2
USER23   IMS2

(scrolled right to screen 3)
User     MbrName Gbl  LSRM LRcvy    LConvID LConvTran LConvStat LclStat
USER23   IMS1    Y
USER23   IMS1         LCL  CONV,FP
USER23   IMS2         LCL  CONV,FP                               ALLOC,CONVACT
USER23   IMS2                              1 TRAN1A    CONVHELD
USER23   IMS2                              2 TRAN1B    CONVACTV
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information.

USER23 exists on IMS1, IMS2 and in the resource structure. IMS1 displays a global line that shows the user is active and owned on IMS2, and its status recovery mode is LOCAL, which means conversation information is not known globally. IMS1 also displays a local line showing that USER23 exists locally, but has no local status. IMS2 displays the local information for the active user, which includes one status line that shows that an active conversation exists, and an additional output line for each conversation active or held locally.

Example 6 for QUERY USER command

TSO SPOC input:

```
QRY USER NAME(USER23) SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
User     MbrName CC Gbl QCnt EMHQCnt Owner  SRM  Rcvy     Userid   Node   Version#
USER23   IMS1     0 Y      0       0 IMS2   LCL  CONV,FP  UID23    NODE23       38
USER23   IMS1     0
USER23   IMS2     0

(scrolled right to screen 2)
User     MbrName Gbl  AutLNode AutLMdt  AutLDsc  AutLID Status
USER23   IMS1    Y    NODE23   SLU2MOD2 NODE23          ALLOC,RM,RMACTIVE,RMOWNED
USER23   IMS1
USER23   IMS2

(scrolled right to screen 3)
User     MbrName Gbl  LQCnt LSRM LRcvy    LUserid  LNode   LVersion#  LPreset
USER23   IMS1    Y
USER23   IMS1             0 LCL  CONV,FP                            0
USER23   IMS2             0 LCL  CONV,FP  UID23    NODE23         38

(scrolled right to screen 4)
User     MbrName Gbl  LAutLNode LAutLMdt LAutLDsc LAutLID LclStat
USER23   IMS1    Y
USER23   IMS1
USER23   IMS2         NODE23    SLU2MOD2 NODE23           ALLOC,CONVACT
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are active. IMS1, the command master, displays global and local information. IMS2 displays local information.

USER23 exists in IMS1, IMS2, and in the resource structure. IMS1 displays a global line that shows global queue counts and global status from the resource structure. Global status indicates that the user is active on IMS2, and its status recovery mode is LOCAL, which means conversation information is not known globally. IMS1 also displays a local line showing that USER23 exists locally, but has no local status. IMS2 displays the local information for the active user, which shows that conversation status exists locally.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related tasks:

Setting DEADQ status time with the DLQT parameter (Communications and Connections)

Related reference:

/DISPLAY ASMT command (Commands)

/DISPLAY CONV command (Commands)

/DISPLAY STATUS command (Commands)

/DISPLAY USER command (Commands)

Command keywords and their synonyms (Commands)

QUERY USEREXIT command

Use the type-2 QUERY USEREXIT command to display information about the user exits that are defined in the USER_EXITS section of the DFSDFxxx member of the IMS PROCLIB data set. Information about user exits that are not specified in the USER_EXITS section of the DFSDFxxx member is not displayed in the output of the QUERY USEREXIT command.

The output from this command can show the following information:

- A completion code for the line of output
- The IMSplex member name that built the output line
- The exit routine type, as requested in the command
- The exit routine module name
- The number of active instances of the exit routine
- The number of calls to the exit routine since the last refresh
- The entry point for the exit routine
- The total time the exit routine was used since the last refresh
- The load point of the exit routine
- The time that the exit routine module was last refreshed
- The hexadecimal size of the exit routine
- 32 bytes from this exit routine module (starting at offset +04 from the entry point) that is translated into EBCDIC

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 355
- "Usage notes" on page 356
- "Output fields" on page 356
- "Return, reason, and completion codes" on page 357
- "Examples" on page 358

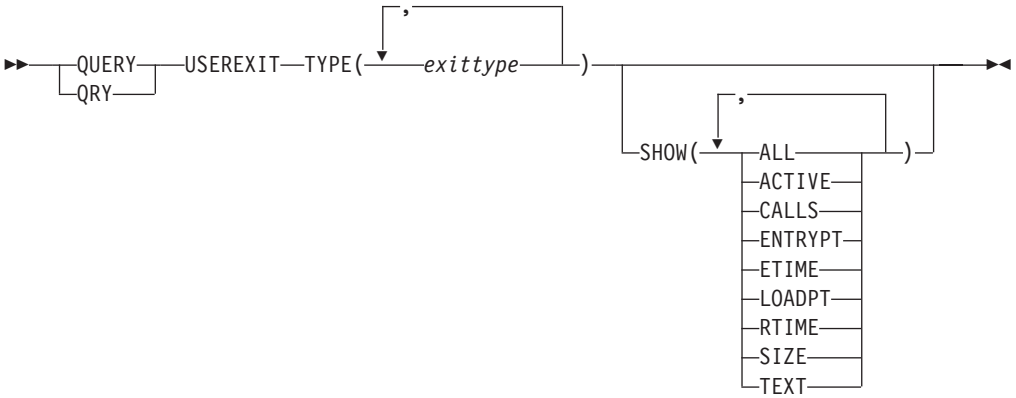
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY USEREXIT command and keywords can be issued.

Table 154. Valid environments for the QUERY USEREXIT command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
QUERY USEREXIT	X	X	X
SHOW	X	X	X
TYPE	X	X	X

Syntax



Keywords

The following keywords are valid for the QUERY USEREXIT command:

TYPE()
Specifies the user exit type or types for which you want information displayed. You can specify a single user exit type or a list of user exit types separated by commas. If the SHOW keyword is not specified, only the names of the exit routines for the specified types are returned. The valid user exit types are:

ICQSEVNT
IMS CQS Event exit type

ICQSSTEV
IMS CQS Structure Event exit type

INITTERM
Initialization/Termination exit type

PPUE
Partner Product exit type

RESTART
Restart exit type

SHOW()
Specifies the information about the user exit routines to be returned in the output fields of the command response. The exit type and module name fields are always returned along with the name of the IMS that created the output for the user exit type and the completion code. The valid fields that can be specified are the following:

ALL
All possible output fields are returned.

ACTIVE
The number of currently active instances of the user exit routine. This is a point-in-time number that represents the number of calls to the user exit that are still in progress and have not returned to IMS.

CALLS
The number of calls to the user exit since the last user exit routine refresh. For performance reasons serialization is not obtained when IMS collects this number. For an exit type that can run multiple instances in parallel, this number is an approximation. The maximum value that this field can

contain is 4,294,967,295 (2^{32}-1). When the call count exceeds this value, the field rolls over and starts again from zero.

ENTRYPT
The entry point address of the user exit routine.

ETIME
The total (cumulative) elapsed time in milliseconds spent in the exit module since it was last refreshed. For performance reasons serialization is not obtained when IMS collects this number. For an exit type that can run multiple instances in parallel, this number is an approximation. The maximum value that can be displayed in this field is 2,147,483,647 (2^{31}-1). If the elapsed time exceeds this value, 2147483647 is displayed.

LOADPT
The address at which the user exit routine was loaded.

RTIME
The local date and time that the user exit routine was last refreshed (or initially loaded, if no refreshes have occurred). The format of the output field is:

```
yyyy-mm-dd hh:mm:ss.th
```

SIZE
The size in bytes of the user exit load routine. This value is displayed in hexadecimal.

TEXT
The 32 bytes starting from offset +04 from the entry point of the exit module, translated to EBCDIC with non-printable characters replaced by periods (.). This is a common location for module identification information. If your user exit routines contain printable identification data at this point in the module, you can use the TEXT option to display that information.

Usage notes

You can issue the QRY USEREXIT command only through the Operations Manager (OM) API.

The output contains an entry for each user exit module within each user exit type specified in the QUERY USEREXIT command. The output of this command is defined in XML and is available to automation programs that communicate with OM.

The QRY USEREXIT command is routed to all IMS systems in the IMSplex as its default routine.

Output fields

The following table shows the QUERY USEREXIT output fields. The columns in the table are:

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 155. Output fields for the QUERY USEREXIT command

Short label	Long label	Keyword	Scope	Meaning
ACTIVE	Active	ACTIVE	LCL	Number of active instances of this exit module.
CALLS	Calls	CALLS	LCL	Number of calls to this user exit module since the last refresh.
CC	CC	N/A	N/A	Completion code for the line of output. The completion code is always returned.
CCTXT	CCText	N/A	N/A	Completion code text that briefly explains the meaning of the nonzero completion code.
ENTRYPT	EntryPt	ENTRYPT	LCL	The entry point of this user exit module.
ETIME	ElapseTime	ETIME	LCL	The total time spent in this user exit module since the last refresh.
LOADPT	LoadPt	LOADPT	LCL	The load point of this user exit module.
MBR	MbrName	N/A	N/A	IMSPLEX member that built the output line. Member name is always returned.
NAME	ModName	N/A	LCL	User exit module name. User exit module name is always returned.
RTIME	RefreshTime	RTIME	LCL	The time this user exit module was last refreshed.
SIZE	ModSize	SIZE	LCL	The size in hexadecimal of this user exit module.
TEXT	ModuleText	TEXT	LCL	32 bytes from this user exit module translated into EBCDIC.
TYPE	ExitType	TYPE	LCL	User exit type requested by the QUERY command. User exit type is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY USEREXIT command are standard for all commands entered through the OM API.

The following table contains the return, reason, and completion codes for the QUERY USEREXIT command. Included in the tables is a brief explanation of the codes.

Table 156. Return and reason codes for the QUERY USEREXIT command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY USEREXIT command completed successfully.
X'00000004'	X'00001010'	The QUERY USEREXIT command is not processed because no user exit routines were found that matched the TYPE parameter.
X'0000000C'	X'00003000'	The QUERY USEREXIT command was successful for at least one user exit type. The QUERY USEREXIT command was not successful for one or more user exit types. The completion code indicates the reason for the error with the user exit type. The completion codes that can be returned by the QUERY USEREXIT command are listed in Table 157.
X'0000000C'	X'00003004'	The QUERY USEREXIT command was not successful for any of the user exit types specified. The completion code indicates the reason for the error with the user exit type. The completion codes that can be returned by the QUERY USEREXIT command are listed in Table 157.
X'00000014'	X'00005004'	The QUERY USEREXIT command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The QUERY USEREXIT command processing terminated because of an internal error.

Errors that are unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains completion codes that can be returned on a QUERY USEREXIT command.

Table 157. Completion codes for the QUERY USEREXIT command

Completion code	Completion code text	Meaning
0		The QUERY USEREXIT command completed successfully for the user exit routine.
10	NO RESOURCES FOUND	None of the user exit types specified are known to the IMS that processed the command. The user exit types might have been typed in error. Confirm that the user exit types are spelled correctly on the command.

Examples

The following are examples of the QUERY USEREXIT command:

Example 1 for QUERY USEREXIT command

TSO SPOC input:

```
QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME)
```

TSO SPOC output:

```
Response for: QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME)
ExitType ModName  MbrName   CC      Calls RefreshTime
INITTERM DFSITRX0 SYS3       0   00000001 2011-05-17 12:54:30.71
INITTERM DFSITRX1 SYS3       0   00000001 2011-05-17 12:54:30.71
INITTERM DFSITRX2 SYS3       0   00000001 2011-05-17 12:54:30.71
```

OM API input:

```
CMD(QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.137 20:03:20.991679</statime>
<stotime>2011.137 20:03:20.992536</stotime>
<staseq>C7C8E8C43F3BF54A</staseq>
<stoseq>C7C8E8C43F718BCA</stoseq>
<rqsttkn1>USRT002 10130320</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>USEREXIT        </kwd>
<input>QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TYPE" llbl="ExitType" scope="LCL" key="YES" len="8"
 dtype="CHAR" align="left" />
<hdr slbl="NAME" llbl="ModName" scope="LCL" key="YES" len="8"
 dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" key="NO" len="4"
 dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" key="NO" len="4" dtype="INT"
 align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CALLS" llbl="Calls" scope="LCL" key="YES" len="10"
 dtype="INT" align="right" />
<hdr slbl="RTIME" llbl="RefreshTime" scope="LCL" key="no" len="22"
 dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>TYPE(INITTERM) NAME(DFSITRX0) MBR(SYS3) CC(   0) CALLS(00000001)
 RTIME(2011-05-17 12:54:30.71) </rsp>
<rsp>TYPE(INITTERM) NAME(DFSITRX1) MBR(SYS3) CC(   0) CALLS(00000001)
 RTIME(2011-05-17 12:54:30.71) </rsp>
<rsp>TYPE(INITTERM) NAME(DFSITRX2) MBR(SYS3) CC(   0) CALLS(00000001)
 RTIME(2011-05-17 12:54:30.71) </rsp>
</cmdrspdata>
</imsout>
 LCM2 xmd_outbf_ptr                                      rea='7F41D340'x
 LCM2 calling FM1                                        rea='0C468100'x
 LCM2 xmd_flags 1                                        rea='61000000'x
 LCM2 xmd_flags 2                     rc='0C468100'x     rea='71000000'x
 LCM2 xmd_flags 3                                        rea='79000000'x
```

```
starting CSLULSKP - 01/24 10:48
num rows / cols                        rc='00000003'x      rea='00000007'x
  ending CSLULSKP - 10/17 10:48
num rows / cols                        rc='00000003'x      rea='00000006'x
starting cslulsrt - 5/03  11:24                            rea='00000002'x
starting cslulSRX
  ending cslulSRX
  ending cslulsrt
starting CSLULER1 - - - - - - -                            rea='0C468100'x
 getting ret codes  - - -              rc='0C46EB22'x      rea='00000008'x
 getting rsn codes  - - -              rc='0C46EB34'x      rea='00000008'x
 got ret/rea codes  - - -              rc='00000000'x      rea='00000000'x
starting CSLULER4 - - - - - - -                            rea='0C468100'x
  searching codes - - - - - - -        rc='00000000'x      rea='00000000'x
  ending CSLULER4 - - - - - - -                            rea='0C468100'x
                                       rc='00000000'x      rea='00000000'x
  ending CSLULER1 - - -  -             rc='00000000'x      rea='00000000'x
  exiting CSLULCM2                     rc='00000000'x      rea='00000000'x
 exiting CSLULCMD                      rc='00000000'x
    in selct_panel
starting csluldsp - 5/15  08:59                            rea='0C468100'x
 cscr_1st_col / cscr_1st_ofst init     rc='00000001'x      rea='00000001'x
 cscr_last_col / cscr_last_ofst init   rc='00000006'x      rea='00000016'x
   out selct_panel
  in cslulrsp - 04/26 -                                    rea='0C468100'x
```

Explanation: Information about Initialization/Termination exits is displayed. The information includes the number of calls to the user exit since it was loaded and the date and time that the user exit routine was loaded.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY USERID command

Use the QUERY USERID command to display information about user IDs across the IMSplex. In this context, a user ID is used for user or terminal security. A user ID is not the same as a user, which is either a dynamic (ETO) user, or a static or dynamic (ETO) ISC subpool user. This command can be specified only through the OM API. It is not valid on an XRF alternate.

Subsections:

- "Environment" on page 361
- "Syntax" on page 361
- "Keywords" on page 361
- "Usage notes" on page 363
- "Similar IMS type-1 commands" on page 364
- "Output fields" on page 364
- "QUERY USERID status" on page 365
- "Return, reason, and completion codes" on page 366
- "Examples" on page 367

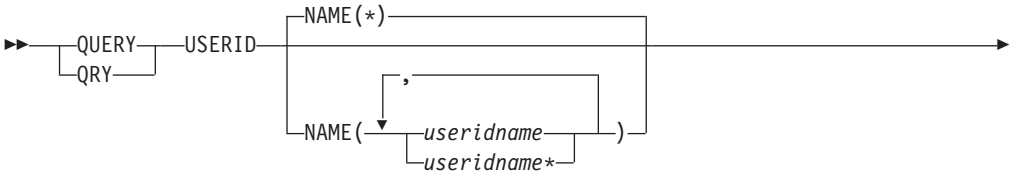
Environment

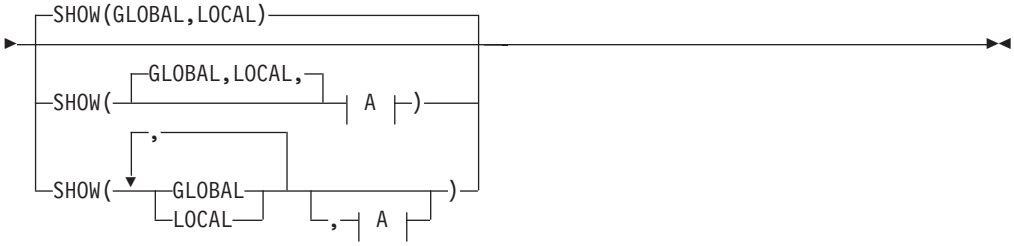
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 158. Valid environments for the QUERY USERID command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY USERID	X		X
NAME	X		X
SHOW	X		X

Syntax





A:

Keywords

The following keywords are valid for the QUERY USERID command:

NAME()
Specifies the names of one or more user IDs that are to be displayed. Valid names are 1-8 characters, and wildcards can be specified. To display all user IDs, specify NAME(*). NAME(*) is the default.

SHOW()
Specifies the user ID output fields to be returned. The user ID is always returned along with the name of the IMS that created the output and the completion code. If SHOW is not specified, only the user IDs are returned. This provides a method for a system management application to obtain a list of all user IDs that are currently known in the IMSplex.

Two parameters, GLOBAL and LOCAL, are used to specify the location (global resources, or local resources) where IMS should obtain the information that is to be displayed. The default is both GLOBAL and LOCAL.

The rest of the parameters are used to specify what information is displayed.

The parameters supported with the SHOW keyword, which can be specified in any order, are:

ALL
Returns all the output fields.

GLOBAL
When GLOBAL is specified, the command master displays global information, depending on the other SHOW parameters specified. This includes information from the resource structure.

The GLOBAL parameter is processed by the command master only. All other IMS systems ignore this parameter. If LOCAL is not also specified, then all IMS systems other than the command master ignore the command.

GLOBAL is applicable only if the command master is using sysplex terminal management.

GLOBAL is not applicable if the command master is not using sysplex terminal management. In this environment, if LOCAL is also specified, then GLOBAL is ignored. Otherwise, the command master rejects the command.

If sysplex terminal management is enabled, then the command master will make requests to RM to determine the appropriate global status.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line, and local information is displayed as another output line.

LOCAL
Specifies that all IMS systems, including the command master, display local information, depending on the other SHOW parameters specified. This includes information local to the IMS processing the command.

The LOCAL parameter is processed by all IMS systems, including the command master.

LOCAL is applicable in any environment, regardless of whether sysplex terminal management is used.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line, and local information is displayed as another output line.

NODE
Displays the node that the user ID is signed on to. This can be a static node or a dynamic node.

OWNER
Displays the owner of the user ID resource in the resource structure. This applies only when sysplex terminal management is enabled, and is processed by the command master only. All other IMS systems ignore this parameter.

The owner is the IMSID (or RSENAME for XRF systems) of the IMS system that owns the user ID. An IMS system owns a user ID resource if the resource is active (the user ID is signed on).

STATUS
Returns local or global status of the user ID. See "QUERY USERID status" on page 365 for a list and meaning of possible status that might be returned.

USER
Displays the dynamic or ISC user associated with the user ID.

VERSION
Displays the RM version number of the user ID resource. This is the version number assigned to the user ID, which is assigned by MVS, and maintained by RM, when the resource is created or updated in the resource structure. VERSION applies only when sysplex terminal management is enabled. VERSION is ignored when sysplex terminal management is not enabled. Only the global version number is displayed. IMS does not maintain the version number in the local system.

Usage notes

The QUERY USERID command can be specified only through the OM API.

The QUERY USERID command is not supported on an XRF alternate system.

The processing of the QUERY USERID command is different depending on whether IMS sysplex terminal management is enabled.

- If IMS sysplex terminal management is not enabled, processing is local for each system. The results of type-1 and type-2 commands are similar.
- If IMS sysplex terminal management is enabled, type-1 and type-2 command processing is similar when displaying local information. However, they differ in how global information is displayed.
- For type-1 /DISPLAY commands with IMS sysplex terminal management enabled, the command master displays information from either the resource structure or the local system, but not both. If the resource being displayed is not owned by any system or is owned by the command master, the command master displays the global resource. However, if the resource is owned by a system other than the command master, the command master displays only the local resource, and the owning system is responsible for displaying the global resource.
- For type-2 QUERY commands with IMS sysplex terminal management enabled, the command master is the only system that displays global resource information, regardless of whether the resource is owned. In addition, the command master displays local resource information. All other IMS systems that process the command display local resource information only. This approach allows more flexibility in displaying all information in an IMSplex.

The SHOW keyword determines which IMS systems process the command, and what information is displayed.

- If SHOW(GLOBAL) is specified, then the command master displays global information, which includes status from the resource structure if sysplex terminal management is enabled (STM=YES defined in DFSDCxxx PROCLIB member). This is true whether or not the user ID is active on any particular IMS system. All other IMS systems to which OM routes the command ignore the

GLOBAL parameter with return code X'00000004' and reason code X'00001000'. However, user ID resources are not maintained in the resource structure if multiple signons are allowed (the same user ID is allowed to sign on to multiple terminals simultaneously). The first IMS that joins the IMSplex determines whether multiple signons are allowed in the IMSplex by specifying SGN=G, M, or Z in the IMS startup parameters.

- If SHOW(LOCAL) is specified, then each IMS system to which OM routes the command (including the command master) processes the command, and displays information that is local to each system.
- If both GLOBAL and LOCAL are specified (which is the default), then the command master displays both global and local information, and all other IMS systems to which OM routes the command display local information.

Similar IMS type-1 commands

The following table shows variations of the QUERY USERID command and the type-1 IMS commands that perform similar functions.

Table 159. Type-1 equivalents for the QUERY USERID command

QUERY USERID command	Similar IMS type-1 command
QUERY USERID SHOW(NODE)	/DISPLAY ASMT USER user /DISPLAY USER user
QUERY USERID SHOW(STATUS)	/DISPLAY USER user
QUERY USERID SHOW(USER)	/DISPLAY ASMT USER user /DISPLAY USER user

Output fields

The following table shows the QUERY USERID output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

SHOW parameter
Identifies the parameter on the SHOW keyword that caused the field to be generated. *Error* appears for output fields that are returned for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field. GBL indicates that the field can be generated only by the command master when displaying global information for SHOW(GLOBAL). LCL indicates that the field can be generated by any IMS displaying local information for SHOW(LOCAL). N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 160. Output fields for the QUERY USERID command

Short label	Long label	SHOW parameter	Scope	Meaning
CC	CC	N/A	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 366 for more information. The completion code is always returned.
CCTXT	CCText	*Error*	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
GBL	Gbl	GLOBAL	GBL	If 'Y', then the output reflects the status found globally in RM. If blank, then the output reflects the status found locally.
LNODE	LNode	NODE	LCL	Identifies the dynamic or static node associated with the user ID on the local system.
LSTT	LclStat	STATUS	LCL	Local user ID status. See "QUERY USERID status" for information about the user ID status that might be returned.
LUSER	LUser	USER	LCL	Identifies the dynamic or ISC user associated with the lterm on the local system.
MBR	MbrName	N/A	N/A	IMSplex member that built output line. IMS identifier of the IMS that built the output. The IMS identifier is always returned.
NODE	Node	NODE	GBL	Identifies the dynamic or static node associated with the user ID in the resource structure.
OWNER	Owner	OWNER	GBL	Resource owner. IMS identifier or RSENAME of IMS where the user ID is active. If no owning IMS system exists and RM contains an entry for the resource, the owner field will be blank.
STT	Status	STATUS	GBL	Global user ID status from the resource structure. See "QUERY USERID status" for information about the user ID status that might be returned.
UID	UserID	N/A	N/A	User ID name. The user ID is always returned.
USER	User	USER	GBL	Identifies the dynamic or ISC user associated with the user ID in the resource structure.
VER	Version#	VERSION	GBL	Version number for the user ID resource being maintained in the resource structure. This field applies only when STM is enabled.

QUERY USERID status

The following table shows the possible user ID status that can be displayed. The columns in the table are:

Status The user ID status that is displayed.

Scope The scope of the status. GBL indicates that the status can be global (it exists in the resource structure when STM is enabled), and is returned with the STT short label. LCL indicates that the status can be local, and is returned with the LSTT short label.

Meaning
Provides a brief description of the status.

Table 161. QUERY USERID status

Status	Scope	Meaning
RM	GBL	The user ID exists in the resource structure managed by RM.
RMACTIVE	GBL	The user ID is active (signed-on) in the IMSplex, as indicated in the RM structure (RM active).
RMOWNED	GBL	The user ID is owned by an IMS system in the IMSplex, as indicated in the RM structure (RM owned).

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY USERID command. The OM return and reason codes that may be returned as a result of the QUERY USERID command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the QUERY USERID command.

Table 162. Return and reason codes for the QUERY USERID command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001000'	The command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000008'	X'00002014'	An invalid character was specified in the resource name.
X'00000008'	X'00002040'	An invalid parameter value was specified. An invalid SHOW value may have been specified.
X'0000000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See Table 163 on page 367 for details.
X'0000000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code. See Table 163 on page 367 for details.
X'00000010'	X'00004004'	Command processing terminated because CQS was not active.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	Command processing terminated because the resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'00004104'	Command processing terminated because RM is not available.

Table 162. Return and reason codes for the QUERY USERID command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004108'	Command processing terminated because SCI is not available.
X'00000014'	X'00005004'	A DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	Command processing terminated because of an RM error.
X'00000014'	X'00005104'	Command processing terminated because of a CQS error.
X'00000014'	X'00005108'	Command processing terminated because of an SCI error.
X'00000014'	X'00005FFF'	Command processing terminated because of an internal IMS error.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 163. Completion codes for the QUERY USERID command

Completion code	Completion code text	Meaning
0		The QUERY USERID command completed successfully for the resource.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
1A3	Userid resource is in error	The user ID resource was found in the resource structure, and an associated resource was needed, but it was either not found or appeared to be in error. This is normally an error condition. However, it could be a temporary condition caused by terminal or command activity. The command should be retried.

Examples

The following are examples of the QUERY USERID command:

Example 1 for QUERY USERID command

TSO SPOC input:

```
QRY USERID NAME(USERID*,XYZ) SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
UserID   MbrName CC CCText              Gbl Owner  Node     User     Version#
USERID01 IMS1     0                     Y   IMS1   NODE01   USER01          1
USERID01 IMS1     0
USERID02 IMS1     0                     Y   IMS2   NODE02   USER02          1
USERID02 IMS2     0
USERID03 IMS1     0                     Y   IMS1   NODE03                   1
USERID03 IMS1     0
XYZ      IMS1    10 NO RESOURCES FOUND  Y
XYZ      IMS1    10 NO RESOURCES FOUND
XYZ      IMS2    10 NO RESOURCES FOUND

(scrolled right to screen 2)
UserID   MbrName Gbl  Status               LNode    LUser
USERID01 IMS1    Y    RM,RMACTIVE,RMOWNED
USERID01 IMS1                              NODE01   USER01
USERID02 IMS1    Y    RM,RMACTIVE,RMOWNED
USERID02 IMS2                              NODE02   USER02
USERID03 IMS1    Y    RM,RMACTIVE,RMOWNED
USERID03 IMS1                              NODE03
XYZ      IMS1    Y
XYZ      IMS1
XYZ      IMS2
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Single-signon for user IDs is enforced. IMS1, the command master, displays global and local information. IMS2 displays local information only.

- USERID01 is signed on to dynamic or ISC node NODE01, user USER01 on IMS1.
- USERID02 is signed on to dynamic or ISC node NODE02, user USER02 on IMS2.
- USERID03 is signed on to static node NODE03 on IMS1.
- XYZ does not exist.

Example 2 for QUERY USERID command

TSO SPOC input:

```
QRY USERID NAME(USERID*) SHOW(ALL)
```

TSO SPOC output:

```
UserID   MbrName CC LNode     LUser
USERID11 IMS1     0 NODE11    USER11
USERID12 IMS1     0 NODE12A   USER12A
USERID12 IMS1     0 NODE12B   USER12B
USERID12 IMS2     0 NODE12C   USER12C
USERID12 IMS2     0 NODE12D
USERID13 IMS2     0 NODE13
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Mutliple-signons for user IDs are enabled, so user ID resources are not maintained in RM. IMS1 displays local information only because user IDs are not maintained in RM. IMS2 displays local information only.

- USERID11 is signed on to dynamic or ISC node NODE11, user USER11 on IMS1.
- USERID12 is signed on to multiple terminals simultaneously: dynamic or ISC nodes NODE12A and NODE12B on IMS1, dynamic or ISC node NODE12C on IMS2, and static node NODE12D on IMS2.
- USERID13 is signed on to static node NODE13.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

- /DISPLAY ASMT command (Commands)
- /DISPLAY USER command (Commands)
- Command keywords and their synonyms (Commands)

Chapter 6. QUEUE commands

Use the IMS QUEUE commands to queue or dequeue LTERMs and transactions.

These commands can be issued to an IMSplex using the Batch SPOC utility.

Subsections:
- "QUEUE LTERM command"
- "QUEUE TRAN command" on page 376

QUEUE LTERM command

Use the QUEUE LTERM command to enqueue a message to the specified LTERM or to dequeue and discard messages currently enqueued to the LTERM.

When IMS uses local queue manager data sets, all IMS systems that receive this command process it.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 372
- "Usage notes" on page 373
- "Output fields" on page 373
- "Return, reason, and completion codes" on page 374
- "Examples" on page 375

Environment

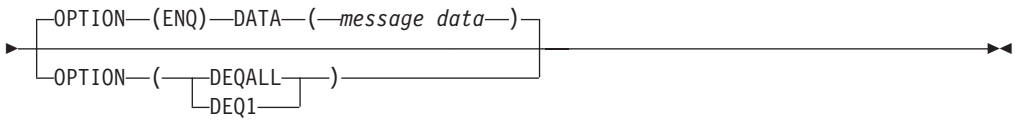
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 164. Valid environments for QUEUE LTERM command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUEUE LTERM	X		X
NAME	X		X
DATA	X		X
OPTION	X		X

Syntax

© Copyright IBM Corp. 1974, 2015



Keywords

The following keywords are valid for the QUEUE LTERM command:

NAME(*lterm***)**
Specifies the 1- to 8-character name of the LTERM.

DATA(*message data***)**
Specifies the message data to enqueue to the LTERM. The data is variable in length. This parameter is valid only for OPTION(ENQ). If the COMM macro is defined with OPTION=NOBLANK, DATA is an optional parameter. If the COMM macro is defined with OPTION=BLKREQD (the default), at least one character, even if it is a blank, must be specified in the message data area.

The message data specified can be in mixed case. If the LTERM is defined as EDIT=ULC, the data is enqueued as specified on the command without any uppercase translation. If the LTERM is defined as EDIT=UC, the data is translated to uppercase before the message is enqueued. No EDIT routine is called before the data is enqueued.

The data is enqueued as a single segment message. The maximum length of data that can be enqueued is 32 763.

OPTION()
Specifies the option parameters.

DEQALL
Indicates that all of the messages currently enqueued to the LTERM should be dequeued and discarded. The dequeue function is similar to a /DEQUEUE LTERM command. The node and user must be stopped before messages can be dequeued. To use the DEQALL option, the LTERM must be defined locally on the IMS system, and the node (if the LTERM is associated to a static node) or the user (if the LTERM is associated to dynamic node) must be stopped.

DEQ1
Indicates the first (oldest) message on the queue for the LTERM should be dequeued and discarded. The dequeue function is similar to a /DEQUEUE LTERM command. The node and user must be stopped before a message can be dequeued. To use the DEQ1 option, the LTERM must be defined locally on the IMS system, and the node (if the LTERM is associated to a static node) or the user (if the LTERM is associated to dynamic node) must be stopped.

ENQ
Indicates that a message is to be enqueued to the specified LTERM. The enqueue function is similar to a /BROADCAST command. If the LTERM is stopped, no message can be enqueued to it.

The QUEUE LTERM OPTION(ENQ) command is not supported for remote LTERMs.

Usage notes

If you use the ENQ option when IMS is using shared message queues, with or without STM=YES, only the command master IMS system processes the command. Messages are enqueued to the shared queues. The LTERM does not need to be defined in the IMS system that processes the command. If the LTERM does not exist, the command master IMS attempts to create the LTERM and associated user structure if ETO is active. If the LTERM is stopped, no message can be enqueued to it. If you use the ENQ option and the LTERM name specified is a remote LTERM, the message is placed on the shared queues using the MSNAME associated with the remote LTERM. The command is not processed if the MSNAME associated with the remote LTERM is stopped.

If you use the DEQALL or DEQ1 option when IMS is using shared message queues and STM=NO, only the command master processes the command. When IMS is using shared message queues and STM=YES, the command is processed by the command master IMS if the user or node is not owned, and by the owning system if the user or node is owned. Messages are dequeued from the shared queues. The node (if LTERM associated to a static node) or the user (if LTERM associated to dynamic node) must be stopped. The LTERM does not need to be defined in the IMS processing the command. If the LTERM does not exist, the IMS processing the command first attempts to create the LTERM and its associated user structure if ETO is active, and then processes the command.

The maximum length of the data that can be enqueued is 32 763 bytes. The data is enqueued to the LTERM as a single segment.

The QUEUE LTERM command includes completion code text with the non-zero completion code. The text can be up to 32 bytes and provides the meaning for the returned completion code:

Note: Before dequeuing one or all messages from shared message queues, stop the user (for ETO terminals) or NODE (or static terminals) for all IMS systems sharing the queues. The LTERM should not be in conversation mode.

Output fields

Output from QUEUE LTERM is returned encapsulated in `<cmdrsphdr>` and `<cmdrspdata>` XML tags. The short label is generated in the XML output. The SHOW keyword listed is the keyword on the command that causes the field to be generated. A value of "n/a" indicates that the output field is always returned. The following table shows the output fields for the QUEUE LTERM command.

Table 165. Output fields for the QUEUE LTERM command

Short label	SHOW keyword	Meaning
CC	n/a	The completion code for the output line. The completion code is always returned.
CCTXT	LCL	The completion code text that briefly explains the meaning of the completion code.
MBR	n/a	The IMSplex member that built the output line. The member name is always returned.
LTERM	n/a	The name of the LTERM resource. The LTERM name is always returned.

Table 165. Output fields for the QUEUE LTERM command (continued)

Short label	SHOW keyword	Meaning
LQCNT	DEQ1, DEQALL	The number of messages dequeued from a local queue.
QNCT	DEQ1, DEQALL	The number of messages dequeued from the shared message queue.

Return, reason, and completion codes

Commands that are issued through the OM API, including QUEUE LTERM, produce a standard set of OM return and reason codes, which are defined in the CSLORR request.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 166. QUEUE LTERM return and reason codes

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUEUE LTERM command completed successfully.
X'00000004'	X'00001000'	The IMS system is not the command master, so the QUEUE LTERM command is not processed on the IMS system.
X'00000008'	X'00002004'	An invalid keyword combination of DATA and OPTION(DEQ1|DEQALL) is specified, so the QUEUE LTERM command is not processed.
X'00000008'	X'00002008'	An insufficient number of keywords is specified, so the QUEUE LTERM command is not processed.
X'0000000C'	X'00003004'	The QUEUE LTERM command is not successful for the resource name specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the command are listed in the QUEUE LTERM completion code table.
X'00000010'	X'0000400C'	The QUEUE LTERM command is invalid on the XRF alternate.
X'00000010'	X'00004014'	The QUEUE LTERM command is invalid on the RSR tracker.
X'00000010'	X'0000401C'	The QUEUE LTERM command is invalid on the FDBR region.
X'00000010'	X'00004034'	The message queues are not available, so the QUEUE LTERM command is not processed.
X'00000014'	X'00005004'	DFSOCMD0 GETBUF storage could not be obtained, so the QUEUE LTERM command is not processed.
X'00000014'	X'00005008'	DFSPOOL HIOP storage could not be obtained, so the QUEUE LTERM command is not processed.

The following table contains completion codes that can be returned on a QUEUE LTERM command.

Table 167. QUEUE LTERM completion codes

Completion code	Meaning
0	The QUEUE LTERM command completed successfully for the specified resource.
10	Because the resource name was not found, the QUEUE LTERM command is not processed.
12	Because the LTERM is active, the QUEUE LTERM OPTION(DEQ|DEQALL) command is not processed.
1A	Because the LTERM is in conversation, the QUEUE LTERM OPTION(DEQ|DEQALL) command is not processed.
5F	Because the LTERM name has invalid characters, the QUEUE LTERM command is not processed.
8A	An unsupported wildcard name was used, so the QUEUE LTERM command is not processed.
8B	A DFSQMGR request error occurred; the QUEUE LTERM command is not processed.
8C	The LTERM is stopped, so the QUEUE LTERM OPTION(ENQ) command is not processed.
8D	The LTERM is not stopped, so the QUEUE LTERM OPTION(DEQ|DEQALL) command is not processed.
93	The QUEUE LTERM command is rejected, because the LTERM name specified is a reserved name.
9F	The QUEUE LTERM OPTION(ENQ) command is not supported for remote LTERMs.

Examples

The following are examples of the QUEUE LTERM command:

Example 1 for QUEUE LTERM command

TSO Input:

```
QUE LTERM NAME(LTERM1) OPTION(ENQ) DATA(MESSAGE1)
```

TSO Output:

```
Lterm    MbrName    CC
LTERM1   SYS3        0
```

Explanation: The QUEUE LTERM command completed successfully and was processed by the command master IMS.

Example 2 for QUEUE LTERM command

TSO Input:

```
QUEUE LTERM NAME(IMSUS01) OPTION(DEQALL)
```

TSO Output:

```
Lterm    MbrName    CC    Qcnt
IMSUS01  IMS2        0       4
```

Explanation: Dequeues all messages on the LTERM queue. The QUEUE LTERM command completed successfully and processed by the command master IMS.

Example 3 for QUEUE LTERM command

TSO Input:

```
QUEUE LTERM NAME(IMSUS01) OPTION(DEQALL)
```

TSO Output:

```
Lterm    MbrName    CC    LQcnt
IMSUS01  IMS2        0        3
IMSUS01  IMS3        0        0
```

Explanation: Dequeues all messages on the LTERM queue. The QUEUE LTERM command completed successfully and is processed by all IMS systems that receive the command.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUEUE TRAN command

Use the QUEUE TRAN command to enqueue a message to the specified transaction or to dequeue and discard messages currently enqueued to the transaction.

The QUEUE TRAN command is defined to OM as ROUTE=ALL. When this command is issued, OM routes the command to all IMS systems that have registered for the command. Depending on the environment, either only the command master IMS processes the command or all IMS process the command. If only the command master processes the command, all other IMS systems that receive the command return with a return and reason code indicating that they are not the IMS command master.

Subsections:

- "Environment"
- "Syntax" on page 377
- "Keywords" on page 377
- "Usage notes" on page 378
- "Output fields" on page 379
- "Return, reason, and completion codes" on page 379
- "Examples" on page 381

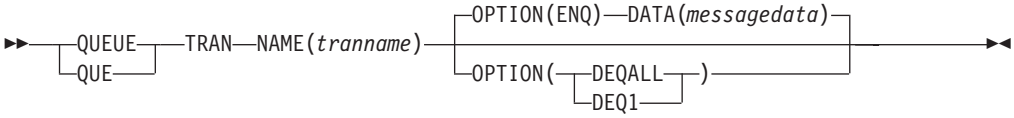
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 168. Valid environments for QUEUE TRAN command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUEUE TRAN	X		X
NAME	X		X
DATA	X		X
OPTION	X		X

Syntax



Keywords

The following keywords are valid for the QUEUE TRAN command:

NAME(*tranname***)**
Specifies the 1- to 8-character name of the transaction.

DATA(*messagedata***)**
Specifies the message data to enqueue to the transaction. The data is variable in length. This parameter is only valid for OPTION(ENQ). If the COMM macro is defined with OPTION=NOBLANK, DATA is an optional parameter. If the COMM macro is defined with OPTION=BLKREQD (the default), at least one character, even if it is a blank, must be specified in the message data area.

The data can be specified in mixed case. If the transaction is defined as EDIT=ULC, the data is enqueued as it was specified on the command, without any uppercase translation. If the transaction is defined as EDIT=UC, the data is translated to uppercase before the message is enqueued. No EDIT routine is called before the data is enqueued.

The data is enqueued as a single segment message. The maximum length of data that can be enqueued is 32 763.

OPTION()
Specifies the option parameters:

ENQ
This is the default. Indicates that a message is to be enqueued to the specified transaction. The ENQ function is processed only by the command master IMS in both the local queues and shared-queues environment. Any output generated by the transaction is sent as an unsolicited output message to the OM that originated the command. Any MFS formatting associated with the output message is not returned.

The transaction specified for the ENQ cannot be full-function response mode or a Fast Path transaction. The initiated transaction can be conversational, but it cannot be in conversational mode with OM.

DEQALL
Indicates that all messages currently enqueued to the transaction should be dequeued and discarded. The transaction must be stopped before messages can be dequeued.

DEQALL is valid in both local queues environment and the shared-queues environment. If IMS is in the local queues environment, all IMS systems that receive the command process the QUEUE TRAN OPTION(DEQALL) command. If IMS is in the shared-queues environment, only the command master IMS processes the QUEUE TRAN OPTION(DEQALL) command.

A QUEUE TRAN OPTION(DEQALL) command can dequeue transaction messages queued to a transaction from APPC or OTMA Clients. The command only dequeues the transaction messages. The APPC or OTMA transaction instance blocks (TIB) created at the IMS system are not deleted and the storage may remain allocated until the next IMS restart. For APPC, the TIB is not released and the APPC conversation (the client) hangs if there is no timeout value specified. The APPCIOT=(x,Y) timeout value for APPC would deallocate the APPC conversation and release the TIB after the timeout limit has been reached.

DEQ1
Indicates that the first (oldest) message on the queue for the transaction should be dequeued and discarded. The transaction must be stopped before the message can be dequeued.

A QUEUE TRAN OPTION(DEQ1) command can dequeue transaction messages queued to a transaction from APPC or OTMA Clients. The command only dequeues the transaction messages. The APPC or OTMA transaction instance blocks (TIB) created at the IMS system are not deleted and the storage may remain allocated until the next IMS restart. For APPC the TIB is not released and the APPC conversation (the client) hangs if there is no timeout value specified. The APPCIOT=(x,Y) timeout value for APPC would deallocate the APPC conversation and release the TIB after the timeout limit has been reached.

Usage notes

This command is processed if the transaction is defined locally at the IMS system. If the transaction is not defined at the IMS system, the Destination Creation exit routine, DFSINSX0, is called. If the exit routine successfully creates a transaction, the command is processed. If the exit does not create a transaction, a response line is returned with an error completion code indicating that the transaction is not found.

The effect of this command differs depending on whether the IMS processing it uses RM's resource structure. If it does use the resource structure, a message can be enqueued even if the transaction is stopped locally if the transaction does not have a global status of STOQ. To dequeue one or all messages, however, the transaction must be stopped globally and locally for scheduling (STOSCHD). If the IMS does not use the resource structure, no messages can be enqueued if the transaction is stopped for queuing (STOQ). To dequeue one or all messages, the transaction must be stopped for scheduling (STOSCHD).

The QUEUE TRAN OPTION(ENQ) command is not supported for conversational, response mode, Fast Path, and remote transactions. This form of the command performs security checking on the transaction name and on the user ID that issued the command. For `OPTION(ENQ)`, workload management (WLM) CLASSIFY calls are issued using information passed to the OM from the command. The information passed includes LUNAME=DFSOMAPI and the user ID. Also for `OPTION(ENQ)`, if the transaction name specified is a remote transaction, the message is placed on the shared queues (on the transaction ready queue) using the remote transaction name.

The command is not processed if the MSNAME associated with the remote transaction is stopped. The TM and MSC Message Routing and Control user exit routine, DFSMSCE0, is not called for a QUEUE TRAN command.

The maximum length of the data that can be enqueued is 32 763 bytes. The data is enqueued to the transaction as a single segment.

If the transaction is processed by an IMS Version 10 or higher system, any generated output to the input destination is sent as an unsolicited output message to the OM that originated the command. The transaction name and IMS ID that processed the transaction are included in the output as follows:

```
8 byte transactionname + 8 byte IMSID + output message
```

If the transaction is processed by an IMS Version 9 or earlier, any generated output to the input destination is not queued. The application program receives a status code of AD.

Output fields

Output from the QUEUE TRAN command is returned encapsulated in `<cmdrsphdr>` and `<cmdrspdata>` XML tags. The short label is generated in the XML output. The keyword listed is the keyword on the command that causes the field to be generated. A value of "n/a" indicates that the output field is always returned. A value of *error* indicates that the output field is returned if there is an error (for example, a nonzero completion code).

Table 169. QUEUE TRAN output

Short label	SHOW keyword	Meaning
CC	n/a	The completion code for the output line. The completion code is always returned.
CCTXT	n/a	The completion code text that explains the completion code. The member name is always returned.
MBR	*error*	The IMSPlex member that built the output line.
TRAN	n/a	The name of the transaction resource.
LQCNT	OPTION(DEQ1 | DEQALL)	The number of messages dequeued from a local queue.
QNCT	OPTION(DEQ1 | DEQALL)	The number of messages dequeued from the shared message queue.

Return, reason, and completion codes

Commands that are issued through the OM API, including QUEUE TRAN, produce a standard set of OM return and reason codes, which are defined in CSLORR.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 170. QUEUE TRAN return and reason codes

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUEUE TRAN command completed successfully.
X'00000004'	X'00001000'	The IMS system is not the command master, so the QUEUE TRAN command is not processed on the IMS system.
X'00000008'	X'00002004'	An invalid keyword combination of `DATA` and `OPTION(DEQ|DEQALL)` is specified, so the QUEUE TRAN command is not processed.
X'00000008'	X'00002008'	An insufficient number of keywords is specified, so the QUEUE TRAN command is not processed.
X'0000000C'	X'00003004'	The QUEUE TRAN command is not successful for the resource name specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the command are listed in the QUEUE TRAN completion code table.
X'00000010'	X'0000400C'	The QUEUE TRAN command is invalid on the XRF alternate.
X'00000010'	X'00004014'	The QUEUE TRAN command is invalid on the RSR tracker.
X'00000010'	X'0000401C'	The QUEUE TRAN command is invalid on the FDBR region.
X'00000010'	X'00004034'	The message queues are not available, so the QUEUE TRAN command is not processed.
X'00000014'	X'00005004'	DFSOCMD0 GETBUF storage could not be obtained, so the QUEUE TRAN command is not processed.
X'00000014'	X'00005008'	DFSPOOL HIOP storage could not be obtained, so the QUEUE TRAN command is not processed.
X'00000014'	X'00005FFF'	The QUEUE TRAN command could not be processed because of an internal error.

The following table contains completion codes that can be returned on a QUEUE TRAN command.

Table 171. QUEUE TRAN completion codes

Completion code	Meaning
0	The QUEUE TRAN command completed successfully for the specified resource.
10	The resource was not found.
17	Another command is in progress for this transaction.
1A	Transaction has an active conversation. Use the /EXIT command to terminate the conversation and to dequeue messages.
51	No resource structure
52	Resource structure full

Table 171. QUEUE TRAN completion codes (continued)

Completion code	Meaning
5F	The QUEUE TRAN command is rejected, because the transaction name specified has invalid characters.
8A	An unsupported wildcard name was used, so the QUEUE TRAN command is not processed.
8B	A DFSQMGR request error occurred, so the QUEUE TRAN is not processed.
8C	The transaction is stopped, so the QUEUE TRAN OPTION(ENQ) is not processed.
8D	The transaction is not stopped, so the QUEUE TRAN OPTION(DEQ|DEQALL) is not processed.
94	RM Request error
96	A RACF security failure has occurred, so the QUEUE TRAN OPTION(ENQ) is not processed.
98	CQS request error
99	Transaction is not initialized
9A	QUEUE TRAN OPTION(ENQ) is not supported for conversational mode.
9B	QUEUE TRAN OPTION(ENQ) is not supported for Fast Path transactions.
9C	QUEUE TRAN OPTION(ENQ) is not supported for response mode transactions.
9D	The MSNAME associated to remote transaction is stopped.
9F	QUEUE TRAN OPTION(ENQ) is not supported for a remote transaction.

Examples

The following are examples of the QUEUE TRAN command:

Example 1 for QUEUE TRAN command

TSO Input:

```
QUEUE TRAN NAME(PART) OPTION(DEQALL)
```

TSO Output:

```
Trancode  MbrName  CC  LQcnt
---------------------------------
PART      IMS1     0   5
PART      IMS2     0   2
```

Explanation: The command was processed by all the IMS systems that received the command. IMS1 had five messages dequeued from its local queue, and IMS2 had two messages dequeued from its local queue.

Example 2 for QUEUE TRAN command

TSO SPOC input:

```
QUEUE TRAN NAME(PART) DATA(message1)
```

TSO SPOC output:

```
Trancode MbrName    CC
PART     IMS2        0
```

Explanation: The QUEUE TRAN command completed successfully and was processed by the command master.

Example 3 for QUEUE TRAN command

TSO SPOC input:

```
QUEUE TRAN NAME(PART) OPTION(DEQALL)
```

TSO SPOC output:

```
Trancode MbrName    CC     Qcnt
PART     IMS3        0        2
```

Explanation: This example shows the QUEUE TRAN OPTION(DEQALL) in a shared-queues environment. Two messages are dequeued by the command master IMS.

Example 4 for QUEUE TRAN command

TSO SPOC input:

```
QUEUE TRAN NAME(PART) OPTION(DEQALL)
```

TSO SPOC output:

```
Trancode MbrName    CC    LQcnt
PART     IMS2        0        2
PART     IMS3        0        0
PART     SYS3        0        1
```

Explanation: This example shows the QUEUE TRAN OPTION(DEQALL) in a Local Queues environment. Each IMS processes the command.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

Chapter 7. /QUIESCE command

The /QUIESCE command initiates the shutdown and deallocates the user for the specified ISC node.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 384
- "Example" on page 384

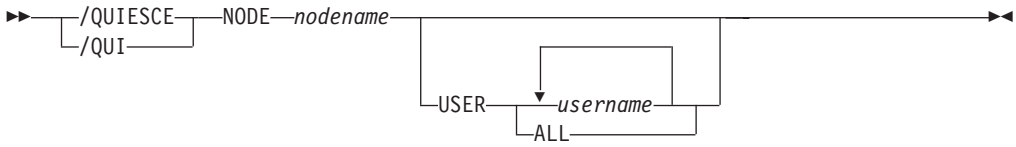
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 172. Valid environments for the /QUIESCE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/QUIESCE	X		X
NODE	X		X
USER	X		X

Syntax



Keywords

The following keywords are valid for the /QUIESCE command:

NODE
Specifies the VTAM node for the user to be shut down and deallocated.

USER
If the USER keyword is omitted, all half-sessions of an ISC node are affected. The half-sessions must be connected.

Restrictions for using NODE and USER parameters together:

- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- /QUIESCE NODE USER commands are valid for ISC nodes only.

© Copyright IBM Corp. 1974, 2015

Usage notes

This command can be issued to an IMSplex using the Batch SPOC utility.

The /QUIESCE NODE command is valid for ISC nodes only.

/QUIESCE resets preset mode, test mode, lock node, lock lterm, pstop lterm, and purge lterm because these statuses are not significant and are not kept after a logon or restart. /QUIESCE also takes other actions depending on the recovery settings for the node:

RCVYCONV=NO
: /QUIESCE causes any IMS conversations (active and held) to be terminated. Any conversational message that is queued or being processed has its output response message delivered asynchronously.

RCVYFP=NO
: /QUIESCE causes Fast Path status and messages to be discarded

RCVYRESP=NO
: /QUIESCE resets full-function response mode.

If global resource information is not kept in Resource Manager, /QUIESCE deallocates the user and resets status locally. If global resource information is kept in Resource Manager, /QUIESCE deallocates the user and resets status globally. If the user has no significant status, /QUIESCE deletes the user in Resource Manager. If the node has no significant status, and there are no other half-sessions for the node, /QUIESCE deletes the node in Resource Manager.

If ROUTE is specified, it should be specified with ROUTE(*). The command fails if not routed to the IMS where the node is active.

Example

Entry ET:

```
/QUIESCE NODE CAL USER LAX
```

Response ET:

```
/DFS058I  QUIESCE COMMAND COMPLETED
```

Explanation: The half-session of node CAL using user LAX is shut down.

Chapter 8. /RCLSDST command

The /RCLSDST (remote close destination) command causes IMS to disconnect the VTAM terminal from which the command is entered.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Example" on page 386

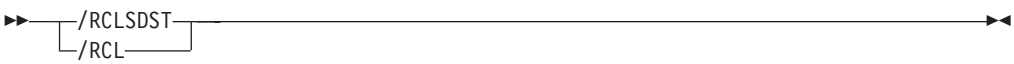
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command can be issued.

Table 173. Valid environments for the /RCLSDST command

Command	DB/DC	DBCTL	DCCTL
/RCLSDST	X		X

Syntax



Usage notes

If you are in an active conversational mode, /EXIT or /HOLD must be entered before /RCLSDST is executed. If this command is issued by a signed on user, the user is signed off.

This command does not reset preset mode.

/RCLSDST resets preset mode, test mode, lock node, lock lterm, pstop lterm, and purge lterm because these statuses are not significant and, therefore, are not kept after logons and restart. /RCLSDST also takes other actions depending on the recovery settings for the node:

RCVYSTSN=NO
: /RCLSDST acts like a /CHANGE NODE COLDSESS command for FINANCE and SLUP nodes by setting the session status to 'cold'. /RCLSDST acts like a /QUIESCE NODE command for ISC (LU6.1) nodes by initiating the shutdown and deallocating the user for the specified node. This action changes the session status to 'cold'. With these actions taken by the /RCLSDST command, the next session initiation request for this node is allowed to again attempt a session cold start. For ETO nodes, the control block structure could be deleted, if no significant status exists.

RCVYCONV=NO
: /RCLSDST causes any held IMS conversations to be terminated. Any

© Copyright IBM Corp. 1974, 2015

conversational message that is queued or being processed has its output response message delivered asynchronously.

RCVYFP=NO
/RCLSDST causes Fast Path status and messages to be discarded.

RCVYRESP=NO
/RCLSDST resets full-function response mode.

If global resource information is not kept in Resource Manager, /RCLSDST logs a node off and resets status locally. If global resource information is kept in Resource Manager, /RCLSDST logs a node off and resets status globally. If the node has no status, /RCLSDST deletes the node in Resource Manager.

Example

Entry ET:

```
/RCLSDST
```

Response ET:

```
DFS058I  RCLSDST COMMAND COMPLETED
```

Explanation: The entering terminal is logged off IMS.

Chapter 9. /RCOMPT command

The /RCOMPT command sets a particular VTAM terminal component to a ready/not ready state.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes" on page 388
- "Example" on page 388

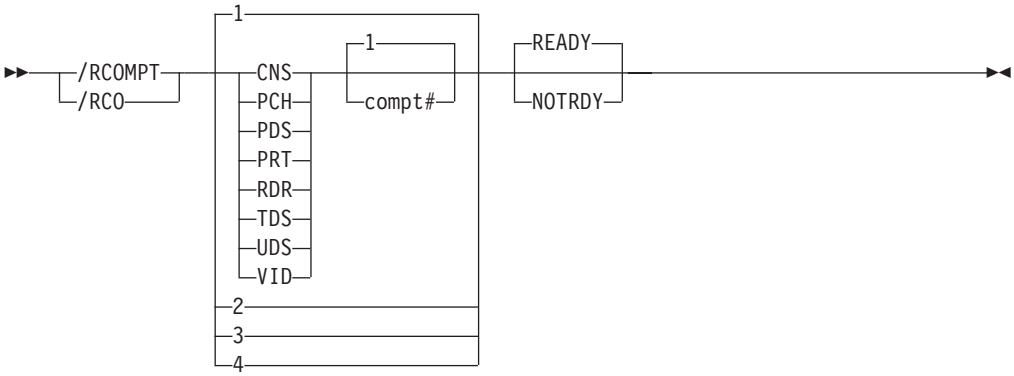
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 174. Valid environments for the /RCOMPT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RCOMPT	X		X
CNS	X		X
NOTRDY	X		X
PCH	X		X
PDS	X		X
PRT	X		X
RDR	X		X
READY	X		X
TDS	X		X
UDS	X		X
VID	X		X

Syntax



© Copyright IBM Corp. 1974, 2015

Usage notes

Output messages queued for a particular component will not be sent unless the component is ready. Depending on terminal type, output operations for other components can continue.

Note: Defaults are READY and 1.

The ready/not ready state set by the /RCOMPT command can be altered by the following:

- Another /RCOMPT command
- A /COMPT, /START, or /RSTART command
- An I/O error on the terminal component

The command format takes one of the following forms:

- A keyword is used.

  A search is made of the components (as defined in the TERMINAL macro during IMS system definition or logon descriptor) for the component defined that corresponds to the specified keyword. When a match is found, that component type is made ready or not ready as specified by the command.
- A keyword is used with a number other than 1 following the keyword.

  The corresponding occurrence of that component type is made ready or not ready, as specified by the command.
- Number 1 through 4 is used instead of a keyword.

  The component affected is the one defined in that position during system definition or logon descriptor independent of component type.

When using ISC, only parameters 1, 2, 3, and 4 are valid.

Example

Entry ET:

```
/RCOMPT VID 2 READY
```

Response ET:

```
DFS058I  RCOMPT COMMAND COMPLETED
```

Explanation: The second display component is declared operable to IMS.

Entry ET:

```
/RCOMPT 4 READY
```

Response ET:

```
DFS058I  RCOMPT COMMAND COMPLETED
```

Explanation: The fourth component defined is declared ready to IMS.

Related reference:

/ASSIGN command (Commands)

Chapter 10. /RDISPLAY command

The /RDISPLAY command refers to the terminal assigned as the master terminal and displays either the logical terminal name and the line and physical terminal numbers, or the logical terminal name and the VTAM NODE name.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Equivalent IMS type-2 commands"
- "Examples" on page 390

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command and keyword can be issued.

Table 175. Valid environments for the /RDISPLAY command and keyword

Command/Keyword	DB/DC	DBCTL	DCCTL
/RDISPLAY	X		X
MASTER	X		X

Syntax

Keywords

The following keyword is valid for the /RDISPLAY command:

MASTER
Specifies the identity of the terminal designated as the master terminal.

Usage notes

If the 3270 master terminal capability was generated during IMS system definition, the logical terminal name, line, and physical terminal number of the secondary master terminal are also displayed.

This command can be issued to an IMSplex using the Batch SPOC utility.

Equivalent IMS type-2 commands

The following table shows variations of the /RDISPLAY command and the IMS type-2 commands that perform similar functions.

© Copyright IBM Corp. 1974, 2015

Table 176. Type-2 equivalents for the /RDISPLAY command

Task	/RDISPLAY command	Similar IMS type-2 command
Displays the primary and secondary master terminal.	/RDISPLAY MASTER	QUERY LTERM STATUS(MTO,SMTO)

Examples

The following are examples of the /RDISPLAY command:

Example 1 for /RDISPLAY command

Entry ET:

```
/RDISPLAY MASTER
```

Response ET:

```
LTERM CNTRL
PTERM 3-1
*91010/123704*
```

Explanation: CNTRL is the master terminal logical terminal and is assigned to LINE 3 PTERM 1.

Example 2 for /RDISPLAY command

Entry ET:

```
/RDISPLAY MASTER
```

Response ET:

```
LTERM CTRL1
PTERM 4-2
LTERM CTRL2
PTERM 4-4
*91010/12370*
```

Explanation: CTRL1 is the primary master terminal logical terminal and is assigned to LINE 4 PTERM 2. CTRL2 is the secondary master terminal logical terminal and is assigned to LINE 4 PTERM 4.

Related reference:

"QUERY LTERM command" on page 100

Chapter 11. /RECOVER commands

The /RECOVER commands are used with the recovery list of database data sets and areas.

In an IMSplex, the /RECOVER command initiates the operation of the Database Recovery facility. OM sends the /RECOVER command to one IMS.

These commands can be issued to an IMSplex using the Batch SPOC utility.

/RECOVER commands are:

- "/RECOVER ADD command"
- "/RECOVER REMOVE command" on page 396
- "/RECOVER START command" on page 400
- "/RECOVER STOP command" on page 405
- "/RECOVER TERMINATE command" on page 408

/RECOVER ADD command

Use the /RECOVER ADD command to add database data sets and areas to a list (recovery list) of database data sets and areas to be recovered using the Database Recovery Facility.

The database data sets and areas can be specified as database data sets, areas, databases, or groups.

Subsections:

- "Environment"
- "Syntax" on page 392
- "Keywords" on page 392
- "Usage notes" on page 395
- "Examples" on page 395

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 177. Valid environments for the /RECOVER ADD command and keywords

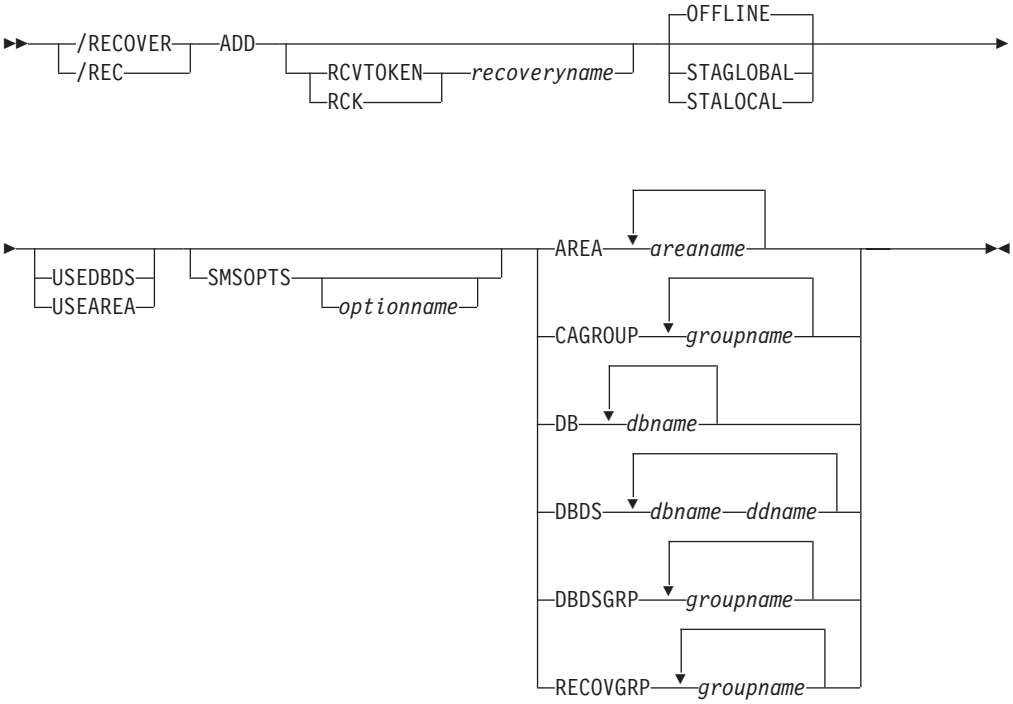
Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
ADD	X	X	
AREA	X	X	
CAGROUP	X	X	
DB	X	X	
DBDS	X	X	
DBDSGRP	X	X	

© Copyright IBM Corp. 1974, 2015

Table 177. Valid environments for the /RECOVER ADD command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
OFFLINE	X	X	
RCVTOKEN	X	X	
RECOVGRP	X	X	
SMSOPTS	X	X	
STAGLOBAL	X	X	
STALOCAL	X	X	
USEAREA	X	X	
USEDBDS	X	X	

Syntax



Keywords

The following keywords are valid for the /RECOVER ADD command:

OFFLINE

With this option, the database remains offline from the IMS system at the completion of the recovery. This option is used to enable the database administrator to verify that the recovery has completed successfully and the databases are ready for use. This is the default if it is not specified on the /RECOVER START command.

The IMS system performing the recovery has exclusive authorization of the database to perform the recovery. The OFFLINE option allows the IMS system to deauthorize the databases at the completion of the recovery without issuing

a START command. This will leave the databases available for processing. If a PITR option has been used in a recovery, the database data set in the RECON is IMAGE COPY NEEDED.

STAGLOBAL
This option is used when the full-function database or databases and Fast Path areas are used in a sysplex data sharing environment. A /START DB command with the GLOBAL option is issued on the IMS system which ran the recovery for all full-function databases affected by recovery. A /START AREA command with the GLOBAL option is used on the IMS system which ran the recovery for all Fast Path areas recovered. This option makes the database or databases available to IMS transactions as soon as all the DBDSs in the recovery list have been recovered. Authorization to use the database is returned to IMS.

STALOCAL
This option is used when the DRF is being executed in the operational IMS system where the database or databases are used. A /START DB command is issued on the IMS system which ran the recovery for all full-function databases affected by recovery. A /START AREA is issued on the IMS system which ran the recovery for all Fast Path areas recovered. Fast Path databases are not started with the /START DB command. Only one option needs to be specified and it applies to all DBDSs and areas added to the recovery list.

USEDBDS | USEAREA
These options are for Fast Path areas, full-function, and HALDB DBDS. When specified, DRF does not need to restore the image copy before applying log updates. You can use this option when you restore image copies prior to recovery.

Note: You must restore non-standard image copies prior to recovery.

This parameter determines whether DRF will restore the image copy for a DBDS before applying log updates.

RCVTOKEN | RCK
Specifies the unique identifier associated with a recovery list that the /RECOVER ADD command operates against. RCVTOKEN is optional. If it is not specified, IMS generates a recovery name. However, if the command is intended to add entries to an existing recovery list, RCVTOKEN must be specified with the intended recovery list token *recoveryname*. When RCVTOKEN is specified in a command, it must come before any keywords that identify the names of database data sets, areas, or groups.

recoveryname
Specifies the unique recovery token associated with the recovery list that the /RECOVER ADD command operates against. This token can be up to eight characters in length.

SMSOPTS
Specifies that the DFSMSdss options are to be associated with the entries being added to the recovery list. The options are only used when restoring image copies created by the Image Copy 2 utility.

optionname
Specifies a unique SMS option.

DELCAT
Specifies that the data set is to be restored using the DFSMSdss optional keyword, DELETECATALOGENTRY.

Attention: This option is to be used with extreme care. DELCAT is required if SMSOPTS is supplied. This option allows you to recover from a scenario where entire volumes are lost but the catalog entries remain. When this option is specified, SMS deletes the prior catalog entries for the database data sets and areas being restored as part of recovery. Before using this option, read the caution under DELETECATALOGENTRY option of the RESTORE command.

AREA
Specifies that one or more Fast Path areas are to be added to a recovery list.

areaname
Specifies a unique Fast Path area.

CAGROUP
Specifies that one or more change accumulation groups, as defined in the RECON data sets, are to have their database data sets and areas added to a recovery list.

groupname
Specifies that the database data sets and areas belonging to the named CA group are to be added to the recovery list.

DB Specifies that all the areas or full-function database data sets for one or more databases are to be added to a recovery list.

dbname
Specifies the database and the associated database data sets or areas that are to be added to a recovery list.

DBDS
Specifies that one or more full-function database data sets are to be added to a recovery list.

dbname ddname
Specifies a full-function database data set is to be added to a recovery list. Full-function database data sets are specified with the /RECOVER ADD DBDS command as an ordered pair. The first member of the pair is the database name. The second member is the DD name. If more than one full-function database data set is specified, the complete ordered pair must be specified for each database data set. All parameters must be separated by at least one blank space.

DBDSGRP
Specifies that one or more DBDS groups as defined in the RECON data sets will have their database data sets and areas added to a recovery list.

groupname
Specifies that the database data sets and areas belonging to the named DBDS group are to be added to the recovery list.

RECOVGRP
Specifies that the listed groups are recovery groups. A recovery group is a group of full-function databases, DEDB areas, or both that the user defines to IMS as related. All DBDSs that make up the full-function databases and all the DEDB areas making up the recover groups specified in the command are added to a recovery list.

groupname
Specifies the unique name of the group whose database data sets and areas are to be added to a recovery list.

Usage notes

Successful completion of a /RECOVER ADD command results in the specified database data sets and areas being added to a recovery list. database data sets and areas can be added to a recovery list by specifying one or more database data sets, databases, change accumulation groups (CAGROUP), database data set groups (DBDSGRP), or recovery groups (RECOVGRP). If a database or group is specified, all database data sets and areas making up the database or group are added to a recovery list. If the specified database is a master database for a partitioned HALDB, all database data sets from all partitions that make up the HALDB are added to a recovery list. All groups (including databases) are defined in DBRC.

Database data sets and areas must be registered with DBRC to be recovered with the IMS Recovery Services. If the database data set, area, or group name is not known to DBRC, it is not added to a recovery list and a message is issued.

If a database data set or area specified in a /RECOVER ADD command (individually or as part of a group) is already on a recovery list, processing for the duplicate is ignored and a message is issued. Other database data sets and areas that are not duplicates are processed normally.

This command can be issued in IMS DBCTL and IMS DB/DC environments.

Examples

The `COMMAND IN PROGRESS` message is issued for /RECOVER ADD commands but is not shown in the following examples.

Example 1 for /RECOVER ADD command

In this example, a /RECOVER ADD STALOCAL command is issued for full-function database data sets. Following a successful recovery of the database data sets, the database is started on the IMS that runs the recovery.

```
/REC ADD STALOCAL DBDS DBNAME1 DDNAME1 DBNAME2 DDNAME2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
```

Example 2 for /RECOVER ADD command

In this example, a /RECOVER ADD command is issued for full-function database data sets. One of the database data sets is not registered in RECON and is rejected.

```
/RECOVER ADD DBDS DBNAME1 DDNAME1 DBNAME1 DDNAME2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6010W UNABLE TO ADD TO RECOVERY LIST, NOT FOUND IN RECON, DBNAME2 DDNAME2
```

Example 3 for /RECOVER ADD command

In this example, a /RECOVER ADD command is issued for full-function database data set. The database that database data set belongs to is still authorized to two IMS systems.

```
/RECOVER ADD DBDS DBNAME1 DDNAME1
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME1 DDNAME1 AUTHORIZED BY IMS1
DFS4299I FRD6003I DBNAME1 DDNAME1 AUTHORIZED BY IMS2
```

Example 4 for /RECOVER ADD command

Databases can be specified as a whole with the /RECOVER ADD DB command. In this example, a full-function database and a Fast Path database have all their database data sets and areas, respectively, added to the recovery list.

```
/REC ADD DB FFDB1 FPDB2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I FFDB1 DDNAME1
DFS4299I FRD6003I FFDB1 DDNAME2
DFS4299I FRD6003I DBAREA3 DDAREA3
DFS4299I FRD6003I DBAREA4 DDAREA4
```

Example 5 for /RECOVER ADD command

If more than one DATAGROUP is specified, the group names must be separated by at least one blank space. In this example, a /RECOVER ADD command is issued for two database groups.

```
/REC ADD DATAGROUP GRPNAME1 GRPNAME2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBNAME4 DDAREA4
DFS4299I FRD6003I DBNAME5 DDNAME5
DFS4299I FRD6003I DBNAME6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBNAME8 DDAREA8
```

Example 6 for /RECOVER ADD command

If one or more RECOVGRP is specified, the group names must be separated by at least one blank. In this example, a /RECOVER ADD command is issued for two recovery groups.

```
/REC ADD RECOVGRP GRPNAME1 GRPNAME2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAMEA DDNAMEA
DFS4299I FRD6003I DBNAMEB DDAREAB
DFS4299I FRD6003I DBNAMEC DDNAMEC
DFS4299I FRD6003I DBNAMED DDAREAD
DFS4299I FRD6003I DBNAMEE DDNAMEE
DFS4299I FRD6003I DBNAMEF DDAREAF
```

/RECOVER REMOVE command

Use the /RECOVER REMOVE command to remove some or all database data sets and areas from the recovery list.

It can only be issued prior to issuing the /RECOVER START command.

Subsections:

- "Environment"
- "Syntax" on page 397
- "Keywords" on page 397
- "Usage notes" on page 399
- "Examples" on page 399

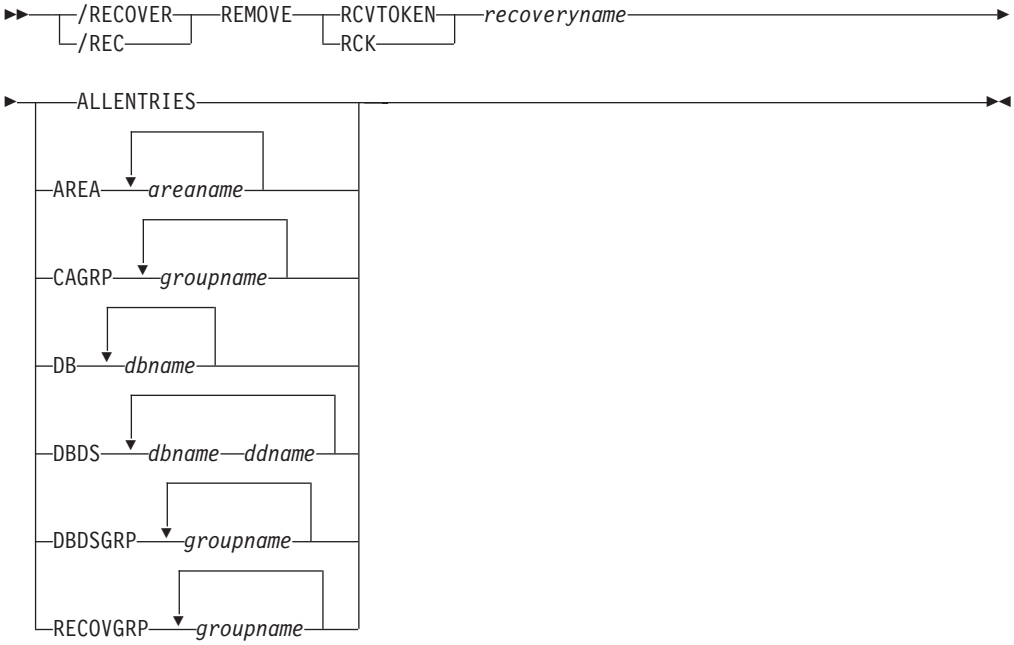
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 178. Valid environments for the /RECOVER REMOVE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
ALLENTRIES	X	X	
AREA	X	X	
CAGROUP	X	X	
DB	X	X	
DBDS	X	X	
DBDSGRP	X	X	
RCVTOKEN	X	X	
RECOVGRP	X	X	
REMOVE	X	X	

Syntax



Keywords

The following keywords are valid for the /RECOVER REMOVE command:

RCVTOKEN | RCK
Specifies the unique identifier associated with the recovery list that the /RECOVER REMOVE command operates against. The RCVTOKEN keyword must come before any keywords that identify the names of database data sets, areas, or groups.

recoveryname
Specifies the unique recovery token associated with the recovery list that the /RECOVER REMOVE command operates against. This token can be up to eight characters in length.

ALLENTRIES
Specifies that the recovery list is to be eliminated.

AREA
Specifies that one or more Fast Path areas are to be removed from the recovery list.

areaname
Specifies a unique Fast Path area.

CAGROUP
Specifies that the database data sets and areas of one or more change accumulation groups as defined in the RECON data sets are to be removed from the recovery list.

groupname
Specifies that the database data sets and areas belonging to a specific CA group are to be removed from the recovery list.

DATAGROUP
Specifies that the database data sets and areas of one or more database groups (as defined in the RECON data sets) are to be removed from the recovery list.

groupname
Specifies the database data sets and areas of the unique group name that are to be removed from the recovery list.

DB Specifies that the full-function database data sets or Fast Path areas making up one or more databases are to be removed from the recovery list.

dbname
Specifies database data sets or areas of the database that are to be added to a recovery list.

DBDS
Specifies that one or more full-function database data sets are to be removed from the recovery list.

dbname
Specifies the database data sets or areas of the database that are to be removed from the recovery list.

ddname
Specifies the DD name of the database data set. If DBDS is specified on the /RECOVER REMOVE command, *dbname* and *ddname* must be specified together.

DBDSGRP
Specifies that the database data sets and areas of one or more DBDS groups as defined in the RECON data sets are to be removed from the recovery list.

groupname
Specifies the database data sets and areas of the group that are to be removed from the recovery list.

RECOVGRP
Specifies that this group is a recovery group. All DBDSs that make up the full-function databases and all the DEDB areas are removed from the recovery list.

groupname
Specifies the database data sets and areas of the database that are to be removed from the recovery list.

Usage notes

Use the /RECOVER STOP command to remove entries after recovery has started.

- If /RECOVER REMOVE is issued before the /RECOVER START command, database data sets and areas specified on the /RECOVER REMOVE command individually or as part of databases or groups are removed from the recovery list. A subsequent /RECOVER START command initiates recovery for the remaining members in the recovery list.
- If a /RECOVER REMOVE ALLENTRIES command is issued before the /RECOVER START command, all elements in the list are removed, and the recovery list is eliminated.
- If the /RECOVER REMOVE command is issued after the /RECOVER START command, the /RECOVER REMOVE command is rejected.

If /RECOVER REMOVE is issued with one or more databases or groups, all database data sets and areas that are part of the database or group specified are removed from the recovery list. If a /RECOVER REMOVE command results in the removal of every data set or area entry from the recovery list, the recovery list is eliminated.

This command executes in IMS DBCTL and IMS DB/DC environments.

Examples

The `command IN PROGRESS` message is issued for /RECOVER commands but is not shown in these examples.

Example 1 for /RECOVER REMOVE command

As with the /RECOVER ADD command, full-function database data sets and Fast Path areas are specified with the /RECOVER REMOVE DBDS command. With the DBDS option, each full-function database data set must be specified as an ordered pair. Each element must be separated by at least one blank space. The first element of the pair is the database name. The second element is the DDNAME. In this example, a /RECOVER REMOVE command is issued for a single full-function database data set.

```
/RECOVER REMOVE RCVTOKEN DFS00001 DBDS DBNAME1 DDNAME1

DFS4299I FRD6016I THE FOLLOWING ENTRIES WERE REMOVED FROM THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
```

If more than one full-function database data set is specified in a /RECOVER REMOVE DBDS command, each dbname/ddname ordered pair must be separated by at least one blank space.

```
/RECOVER REMOVE RCVTOKEN DFS00001 DBDS DBNAME1 DDNAME1 DBNAME3 DDNAME3

DFS4299I FRD6016I THE FOLLOWING ENTRIES WERE REMOVED FROM THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME3 DDNAME3
```

Example 2 for /RECOVER REMOVE command

In this example, a /RECOVER REMOVE command is issued for a single Fast Path area that was not added to the recovery list.

```
/REC REMOVE RCVTOKEN DFS00001 AREA DDAREA1

DFS4299I FRD6018W UNABLE TO REMOVE AREA DDAREA1: NOT IN RECOVERY LIST
```

Example 3 for /RECOVER REMOVE command

In this example, a /RECOVER REMOVE command is issued for a full-function database and Fast Path database. All full-function database data sets and Fast Path areas making up the two databases are removed from the recovery list.

```
/REC REMOVE RCVTOKEN DFS00001 DB FFDB1 FFDB2

DFS4299I FRD6016I THE FOLLOWING ENTRIES WERE REMOVED FROM THE RECOVERY LIST:
DFS4299I FRD6003I FFDB1 DDNAME1
DFS4299I FRD6003I FFDB1 DDNAME2
DFS4299I FRD6003I DBAREA3 DDAREA3
DFS4299I FRD6003I DBAREA4 DDAREA4
```

Example 4 for /RECOVER REMOVE command

In this example, a /RECOVER REMOVE command is issued to stop recovery for the entire recovery list.

```
/REC REMOVE RCVTOKEN DFS00001 ALLENT

DFS4299I FRD6016I THE FOLLOWING ENTRIES WERE REMOVED FROM THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBNAME5 DDNAME5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBAREA7 DDAREA7
DFS4299I FRD6003I DBAREA8 DDAREA8
DFS4299I FRD6017I RECOVERY LIST IS NOW EMPTY
```

Related reference:

"/RECOVER START command"

/RECOVER START command

Use the /RECOVER START command to start the recovery process for all the members of a recovery list.

The recovery process includes performing the following tasks:

- Image copies are restored to the database data sets and areas in the recovery list.
- Change accumulation data is applied to the database data sets and areas in the recovery list.
- The database data sets and areas are brought up to date by applying data changes from log data sets (or up to the recovery time).
- Log data sets cached to a VTS are staged to DASD as a user option.

Subsections:

- "Environment" on page 401
- "Syntax" on page 401
- "Keywords" on page 401
- "Usage notes" on page 403
- "Examples" on page 403

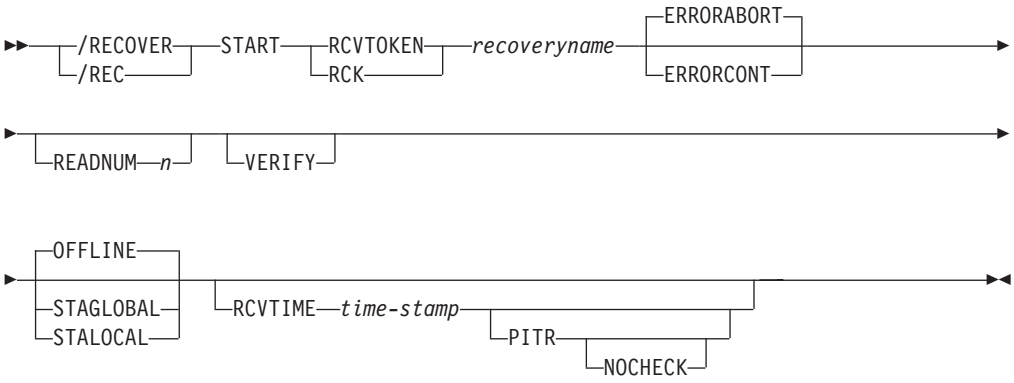
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 179. Valid environments for the /RECOVER START command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
ERRORABORT	X	X	
ERRORCONT	X	X	
NOCHECK	X	X	
OFFLINE	X	X	
PITR	X	X	
RCVTIME	X	X	
RCVTOKEN	X	X	
READNUM	X	X	
REMOVE	X	X	
STAGLOBAL	X	X	
STALOCAL	X	X	
START	X	X	
VERIFY	X	X	

Syntax



Keywords

The following keywords are valid for the /RECOVER START command:

RCVTOKEN | RCK
Specifies the token of the recovery list that will be processed. When RCVTOKEN is specified in a command, it must come before any keywords that identify the names of database data sets, areas, or groups.

recoveryname
Specifies the unique recovery token associated with the recovery list to be processed. This token can be up to eight characters in length.

ERRORABORT
Specifies that recovery stops for all entries in the recovery list if recovery of any database data set can not be completed. ERRORABORT is the default.

ERRORCONT
Specifies that recovery is to continue if recovery is able to complete processing for any database data set.

READNUM *n*
Specifies the number of input devices used in parallel during recovery. Image copies are restored using the number of input devices specified by *n*. When image copies are restored, log data sets are read using the number of input devices specified by *n*.

VERIFY
Specifies the report only execution and obtains a list of the log, change accumulation, and image copy data sets required to process recovery for the associated recovery list. This option allows users to perform set up procedures before invoking the recovery process.

OFFLINE
This option leaves all the databases offline after the recovery is complete. When the recovery is complete, DRF will deauthorize the databases. This leaves the databases in a state that allows normal DBRC processing.

STAGLOBAL
This option is used in a sysplex data sharing environment or with two IMS systems sharing data on the same central processing complex (CPC). A /START DB command with the GLOBAL option is issued internally for DL/I databases and a /START AREA command with the GLOBAL option is issued for the Fast Path areas. OFFLINE, STALOCAL, or STAGLOBAL options specified on the /RECOVER START command are used only for database data sets and areas that were added and did not have any of those options specified.

STALOCAL
This option is used to start the databases on the IMS system that ran the DRF recovery. A /START DB command with the LOCAL option is issued internally.

RCVTIME
Specifies the time stamp to which a point in time or time stamp recovery is to be performed.

time-stamp
The time stamp must have a format that is recognizable to IMS. Note that the UTC offset portion of a time stamp cannot be specified using a symbolic value in this command.

Additionally, the time stamp must by surrounded by single quotation marks ('). For example:

```
/RECOVER START RCVTOKEN R1 RCVTIME '022671213156'
```

PITR
Specifies that a time stamp recovery (TSR) will be performed to the time specified with the RCVTIME parameter regardless if there are any active database allocations for the specified database data sets.

NOCHECK
If a portion of the database data sets making up a database are in the recovery list being started, NOCHECK specifies that the Database Recovery Facility will

not stop a time stamp recovery or a time stamp recovery to any prior point in time (PITR) if one of the following situations occurs:

- All members of the recovery group are not in the same recovery list.
- All members of the recovery group are not being recovered to an equivalent point in time.

Usage notes

Only one /RECOVER START command is allowed to execute in one IMS at a time. If DRF is to run in conjunction with multiple IMS systems simultaneously, ensure that log contention situations do not occur. To avoid log contention situations, ensure that recovery instances that would read the same log data sets do not execute simultaneously in multiple IMS systems.

You can choose to automatically /START any or all members of the recovery list after successful completion of recovery, either on all IMS systems on which they are defined or just the one where the recovery is executed.

If ERRORABORT is in effect, the recovery list will not start until all the DBDSs in the list can be authorized for recovery. This is not true if the ERRORCONT parameter is specified on the /RECOVER START command. Instead, the recovery will continue.

If coordinated online change removes from the system any database data set or area that has been previously added to a recovery list, message `DFS4266I` with reason code NOT FOUND will be issued after the /RECOVER START command for that recovery list is entered.

Examples

The `command IN PROGRESS` message is issued for /RECOVER commands but is not shown in the following examples.

Example 1 for /RECOVER START command

In this example, /RECOVER START initiates recovery for the database data sets and areas from previous examples. Recovery continues until it completes or until one of the database data sets or areas is operable.

```
/REC START RCVTOKEN RCVTKN1 ERRORCONT
DFS4299I FRD6021I RECOVERY STARTED FOR:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
```

Example 2 for /RECOVER START command

In this example, /RECOVER START RCVTOKEN initiates recovery for the database data sets and areas owned by the recovery token RCVTKN2.

```
/REC START RCVTOKEN RCVTKN2
DFS4299I FRD6021I RECOVERY STARTED FOR:
DFS4299I FRD6003I DBNAME1 DDNAME1
```

```
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
```

Example 3 for /RECOVER START command

In this example, /RECOVER START RCVTOKEN OFFLINE READNUM 6 initiates recovery for the database data sets and areas from previous examples. Recovery will not continue if any error is detected for any member of the recovery list. The database data sets and areas remain offline after recovery completes.

```
/RECOVER START RCVTOKEN RCVTKN2 OFFLINE READNUM 6
DFS4299I FRD6021I RECOVERY STARTED FOR:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
```

Example 4 for /RECOVER START command

In this example, /RECOVER START RCVTOKEN ERRORCONT RCVTIME *time-stamp* is issued. TSR continues until it completes or until one of the database data sets or areas undergoing recovery encounters an error. After recovery completes, a message is issued listing each database data set and area successfully recovered.

```
/REC START RCVTOKEN RCVTKN2 ERRORABORT RCVTIME '020011015257' NOCHECK
DFS4299I FRD6021I RECOVERY STARTED FOR RCVTKN2, ERRORABORT, TSR
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
    ...
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME1 DDNAME1
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME2 DDNAME2
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA1 DDAREA1
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME3 DDNAME3
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA4 DDAREA4
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA5 DDAREA5
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA6 DDAREA6
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME7 DDNAME7
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA8 DDAREA8
DFS4277I RECOVERY COMPLETE FOR:  RCVTKN2
```

Example 5 for /RECOVER START command

In this example, /RECOVER START RCVTOKEN RCVTIME*time-stamp* PITR is issued. Point-in-time recovery will continue until it completes or until one of the

database data sets encounters an error. A message is issued listing the database data sets and areas that were not in the recovery list, but might need recovery using point-in-time recovery.

```
/REC START RCVTOKEN RCVTKN2 RCVTIME '020011015257' PITR
DFS4299I FRD6021I RECOVERY STARTED FOR RCVTKN2, ERRORCONT, PITR
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
    ...
DFS4299I FRD6024A GROUP MEMBER DBNAME9 DDNAME9 NOT IN RECOVERY LIST: MEMBER OF A GROUP
DFS4299I FRD6024A GROUP MEMBER DBNAMEA DDNAMEA NOT IN RECOVERY LIST: MEMBER OF A GROUP
DFS4299I FRD6024A GROUP MEMBER DBAREAB DDAREAB NOT IN RECOVERY LIST: MEMBER OF A GROUP
DFS4299I FRD6024A GROUP MEMBER DBNAMEC DDNAMEC NOT IN RECOVERY LIST: MEMBER OF A GROUP
    ...
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME1 DDNAME1
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME2 DDNAME2
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA1 DDAREA1
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME3 DDNAME3
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA4 DDAREA4
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA5 DDAREA5
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA6 DDAREA6
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME7 DDNAME7
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA8 DDAREA8
DFS4277I RECOVERY COMPLETE FOR:  RCVTKN2
```

Related reference:

"/RECOVER REMOVE command" on page 396

"/RECOVER STOP command"

Examples for the DFSURDB0 utility (Database Utilities)

/RECOVER STOP command

Use the /RECOVER STOP command to stop recovery for all database data sets and areas on the recovery list.

Subsections:

- "Environment"
- "Syntax" on page 406
- "Keywords" on page 406
- "Usage notes" on page 407
- "Examples" on page 407

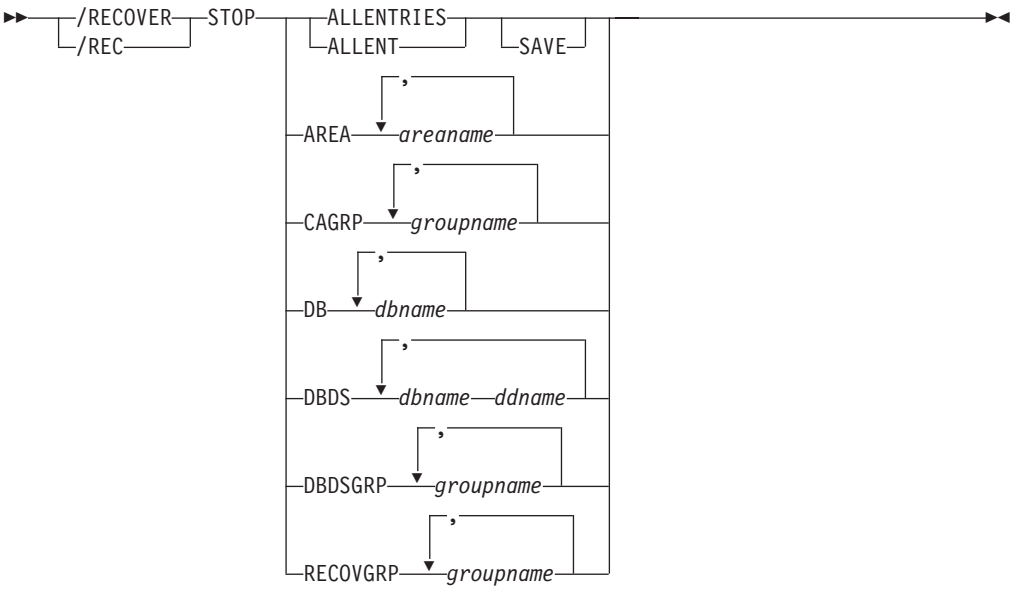
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 180. Valid environments for the /RECOVER STOP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
STOP	X	X	

Syntax



Keywords

The following keywords are valid for the /RECOVER STOP command:

ALLENTRIES
Specifies that recovery is to be aborted for all database data sets and areas (all entries) in the recovery list.

SAVE
Specifies that the recovery list is not to be deleted when recovery is stopped. This parameter is only allowed with the ALLENT parameter after recovery has been initiated with the /RECOVER START command.

AREA
Specifies that recovery processing is to be stopped for the specified Fast Path areas.

areaname
Specifies a unique Fast Path area.

CAGROUP
Specifies that recovery processing is to be stopped for the database data sets and areas making up the specified change accumulation groups as defined in the RECON data sets.

groupname
Specifies the unique name of the group whose database data sets and areas are to have recovery processing be stopped.

DB Specifies that recovery processing is to be stopped for the full-function database data sets or Fast Path areas that make up the specified databases.

dbname
Specifies the database whose database data sets or areas are to be added to a recovery list.

DBDS
Specifies that recovery processing is to be stopped for the specified full-function database data sets.

dbname
Specifies the database whose database data sets or areas are to be added to a recovery list.

ddname
The 8 character identifier associated with the data set name and data set characteristics. *dbname* and *ddname* must be specified together if DBDS is specified on the /RECOVER REMOVE command.

DBDSGRP
Specifies that recovery processing is to be stopped for the database data sets and areas making up the specified DBDS groups as defined in the RECON data sets.

groupname
Specifies the unique name of the group whose database data sets and areas are to have recovery processing be stopped.

RECOVGRP
Specifies that this group is a recovery group. A recovery group is a group of full-function databases or DEDB areas that are considered to be related. All DBDSs that make up the full-function databases and all the DEDB areas are removed from the recovery list.

groupname
Specifies the unique name of the group whose database data sets and areas are to have recovery processing be stopped.

Usage notes

The command can only be issued for a recovery list that has had /RECOVER START issued against it. If the /RECOVER STOP is issued before the /RECOVER START command, it is rejected. If it is issued after the /RECOVER START command, recovery is stopped for all database data sets in the recovery list. After a /RECOVER STOP command successfully processes, subsequent /RECOVER STOP commands are rejected.

If /RECOVER STOP ALLENT is issued, all recovery processing for the affected recovery list halts, and the existing recovery list is deleted.

Examples

The `COMMAND IN PROGRESS` message is issued for /RECOVER commands but is not shown in the following examples.

Example 1 for /RECOVER STOP command

In this example, a /RECOVER STOP command is issued to stop recovery for the entire recovery list.

```
/REC STOP ALLENT
DFS4299I FRD6032I THE FOLLOWING ENTRIES WILL HAVE RECOVERY STOPPED:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
```

```
DFS4299I FRD6003I DBNAME5 DDNAME5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBAREA7 DDAREA7
DFS4299I FRD6003I DBAREA8 DDAREA8
DFS4299I FRD6033I ALL ENTRIES IN RECOVERY LIST, ARE BEING STOPPED
```

Example 2 for /RECOVER STOP command

In this example, a /RECOVER STOP ALLENT SAVE is issued after a /RECOVER START command.

```
/REC STOP ALLENT SAVE
DFS4299I FRD6032I THE FOLLOWING ENTRIES WILL HAVE RECOVERY STOPPED:
DFS4299I FRD6003I DBNAME1 DDNAME1
```

Example 3 for /RECOVER STOP command

In this example, a /RECOVER STOP ALLENT command is issued with no recovery in progress.

```
/REC STOP ALLENT
DFS4299I FRD6031E UNABLE TO STOP ALLENT: RECOVERY NOT IN PROGRESS
```

Related reference:

"/RECOVER START command" on page 400

"/RECOVER TERMINATE command"

/RECOVER TERMINATE command

Use the RECOVER TERMINATE command to delete all lists in BEING BUILT status and to terminate the DRS address space.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes" on page 409
- "Example" on page 409

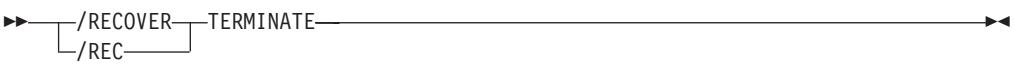
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 181. Valid environments for the /RECOVER TERMINATE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
TERMINATE	X	X	

Syntax



Usage notes

If a recovery is in progress, the recovery will ignore the /RECOVER TERMINATE command and continue to process. When it completes, another /RECOVER TERMINATE command is required. It will not take effect automatically. If you would like to force DRF down while a recovery is running, you must issue the /RECOVER STOP ALLENT command first. This will stop the recovery, and then you can enter the /RECOVER TERMINATE command to terminate the DRF address space.

Example

In this example, a /RECOVER TERMINATE command is issued with no recovery in progress.

```
/RECOVER TERMINATE
DFS4299I FRD4202I DATABASE RECOVERY DATA MANAGER TERMINATION COMPLETE
```

Related reference:

"/RECOVER STOP command" on page 405

Chapter 12. /RELEASE command

The /RELEASE command resumes a conversation that was previously saved by means of the /HOLD command.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Example" on page 412

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command and keyword can be issued.

Table 182. Valid environments for the /RELEASE command and keyword

Command / Keyword	DB/DC	DBCTL	DCCTL
/RELEASE	X		X
CONVERSATION	X		X

Syntax

Keywords

The following keywords are valid for the /RELEASE command:

CONVERSATION
Specifies the 4-digit identification (including leading zeros) of the conversation to be resumed; CONVERSATION conv# is the 4-digit identification conv# that was provided when the conversation was previously held.

Usage notes

The last message sent to the terminal before /HOLD was entered is sent to the terminal again.

/RELEASE is not valid from an LU 6.2 device. LU 6.2 communications cannot release a conversation, whether started by itself or by another communications protocol.

If global resource information is kept in Resource Manager, /RELEASE updates the conversation globally in Resource Manager. If global resource information is not kept in Resource Manager, /RELEASE updates the conversation locally.

© Copyright IBM Corp. 1974, 2015

Example

Entry ET:

```
/RELEASE CONVERSATION 0001
```

Response ET:

IMS does not respond to this command except to resend the last message.

Explanation: Conversation 0001 has been released and can be resumed by the terminal operator.

Chapter 13. /RESET command

The /RESET command eliminates the preset mode established by the /SET command.

Subsections:

- "Environment"
- "Syntax"
- "Example"

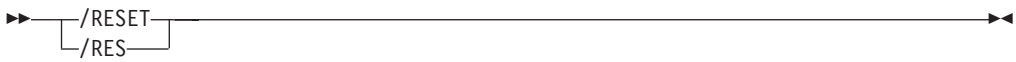
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command can be issued.

Table 183. Valid environments for the /RESET command

Command	DB/DC	DBCTL	DCCTL
/RESET	X		X

Syntax



Example

Entry ET:

```
/RESET
```

Response ET:

```
DFS058I  RESET COMMAND COMPLETED
```

Explanation: The preset mode is no longer in effect.

© Copyright IBM Corp. 1974, 2015

Chapter 14. /RMxxxxxx commands

The /RMxxxxxx commands are multisegment commands that call functions of IMS Database Recovery Control (DBRC). Each /RMxxxxxx command is an online version of the corresponding batch DBRC command.

The following table lists the DBRC commands that are supported online and describes the utility function of each command:

Table 184. Functions of the DBRC commands supported online

Command	Utility function
/RMCHANGE	Changes or modifies information in the RECON data set
/RMDELETE	Deletes information in the RECON data set
/RMGENJCL	Generates JCL for: • IMS Change Accumulation utility • IMS Log Archive utility • IMS Log Recovery utility • IMS Database Image Copy utility • Database Image Copy 2 • IMS Online Database Image Copy utility • Database Recovery utility • User-defined output
/RMINIT	Creates records in the DBRC RECON data set
/RMLIST	Lists information contained in the RECON data set
/RMNOTIFY	Notifies DBRC about additional information, which gets recorded in the RECON data set.

These commands enable the IMS master terminal operator or an authorized terminal operator to run certain DBRC utility functions online. Output is limited to what can be put in a 4 KB buffer (32 KB for /RMLIST).

Subsections:

- "Environment"
- "Syntax" on page 416
- "Keywords" on page 416
- "Usage notes" on page 417
- "Examples" on page 418

Environment

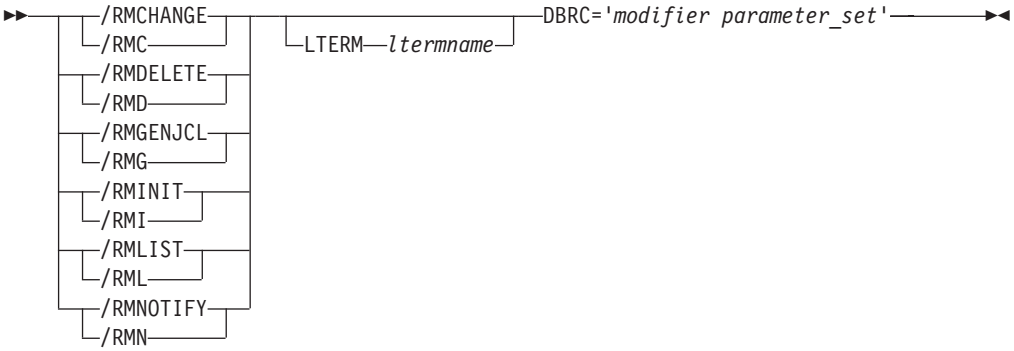
The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command and keyword can be issued.

Table 185. Valid environments for the /RMxxxxxx command and keyword

Command / Keyword	DB/DC	DBCTL	DCCTL
/RMxxxxxx	X	X	X
LTERM	X		X

© Copyright IBM Corp. 1974, 2015

Syntax



Keywords

The following keywords are valid for the /RMxxxxxx command:

LTERM *ltermname*
Specifies the logical terminal designated for output. If you omit the LTERM keyword, the output destination is the input terminal.

Recommendation: Because some of the DBRC commands generate a large amount of output, especially the /RMGENJCL and /RMLIST commands, direct the output to a printer.

DBRC='*modifier parameter_set***'**
Specifies the DBRC modifier for the function specified and the parameters that will be passed to DBRC. The DBRC= parameter is required on all /RMxxxxxx commands.

modifier
The DBRC modifier for the function specified. The following table lists the DBRC modifiers and the /RMxxxxxx commands with which the modifiers can be issued.

Table 186. DBRC modifiers for the /RMxxxxxx commands

	/RMxxxxxx commands					
Modifier	**CHANGE**	**DELETE**	**GENJCL**	**INIT**	**LIST**	**NOTIFY**
ADS	X	X		X		
ALLOC		X				X
ARCHIVE			X			
BKOUT	X	X			X	X
CA	X	X	X	X		X
CAGRP	X	X		X	X	
CLOSE			X			
DB	X	X		X	X	
DBDS	X	X		X	X	
DBDSGRP	X	X		X	X	
GSG		X		X	X	

Table 186. DBRC modifiers for the /RMxxxxxx commands (continued)

	/RMxxxxxx commands					
Modifier	**CHANGE**	**DELETE**	**GENJCL**	**INIT**	**LIST**	**NOTIFY**
HISTORY					X	
IC	X	X	X	X		X
LOG		X			X	
OIC			X			
PART				X		
PRILOG	X					X
RECON	X				X	
RECOV		X	X			X
REORG		X				X
SECLOG	X					X
SG	X	X		X		
SUBSYS	X	X			X	X
UIC	X	X				X
USER			X			

parameter_set
Represents the required and optional parameters that will be passed to DBRC. For an explanation of the values that can be specified, see the corresponding command. For example, if you specify /RMLIST LTERM(*ltermname*) DBRC='DBDS *parameter_set*', see the description of the LIST.DBDS command to understand what parameters can be used in place of *parameter_set*.

Table 187. /RMxxxxx commands and corresponding DBRC commands

Command	Corresponding DBRC command
/RMCHANGE	For the parameters that can be issued with the /RMCHANGE command, see CHANGE commands (Commands).
/RMDELETE	For the parameters that can be issued with the /RMDELETE command, see DELETE commands (Commands).
/RMGENJCL	For the parameters that can be issued with the /RMGENJCL command, see GENJCL commands (Commands).
/RMINIT	For the parameters that can be issued with the /RMINIT command, see INIT commands (Commands).
/RMLIST	For the parameters that can be issued with the /RMLIST command, see LIST commands (Commands).
/RMNOTIFY	For the parameters that can be issued with the /RMNOTIFY command, see NOTIFY commands (Commands).

Usage notes

All /RMxxxxxx formats require an EOM indication to denote end-of-message. An EOS indication must be included for all segments, if any, that precede the last segment. If comments are included with the /RMxxxxxx commands, they must be enclosed in asterisks.

If a failure other than the loss of both RECON data sets occurs while DBRC is processing an online command, DBRC makes the command unavailable for the remaining time the IMS online region is running. After determining and correcting the cause of the original failure, the command can be made available again by resubmitting the online command with the RESET parameter specified in the parameter set. It is the verb, rather than the modifier, level of the command that DBRC makes unavailable. That is, if a DBRC INIT.DB command fails, DBRC makes *all* INIT commands unavailable. DBRC sends an error message to the originating terminal when the command fails. You can still issue the failing command from other IMS online regions.

Note: DBRC does not make GENJCL commands unavailable, because the GENJCL.ARCHIVE command is needed for automatic archiving.

These commands can be issued to an IMSplex using the Batch SPOC utility.

DBRC does not remember command failures across IMS restarts because it assumes that you will correct the error before restarting IMS.

Examples

The following are examples of the /RMxxxxxx commands:

Examples for /RMCHANGE command

Entry ET (with comments):

```
/RMCHANGE DBRC='DB DBD(DIVNTZ04) SHARELVL(3)'. *COMMENT*.
```

Response ET:

```
CHANGE.DB DBD(DIVNTZ04) SHARELVL(3)
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMC COMMAND COMPLETED
```

Entry ET:

```
/RMCHANGE DBRC='DBDS DBD(DIVNTZ04) DDN(DBHVSAM1) ICON'.
```

Response ET:

```
CHANGE.DBDS DBD(DIVNTZ04) DDN(DIVNTZ04) ICON
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMC COMMAND COMPLETED
```

Entry ET:

```
/RMC DBRC='DBDS DBD(DEDBJN21) AREA(DB21AR7) VSO PREOPEN'.
```

Response ET:

```
DFS000I CHANGE.DBDS DBD(DEDBJN21) AREA(DB21AR7) VSO PREOPEN
DFS000I DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
```

Example for /RMDELETE command

Entry ET (with comments):

```
/RMDELETE DBRC='DB DBD(DIVNTZ04)'. *COMMENT*.
```

Response ET:

```
DELETE.DB DBD(DIVNTZ04)
DSP0203I COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMD COMMAND COMPLETED
```

Example for /RMGENJCL command

Entry ET (with comments):

```
/RMGENJCL LTERM SMITH DBRC='IC DBD(HDAMVSAM) DDN(DD1) LIST'.
```

Response ET:

```
DSP058I RMG COMMAND COMPLETED
```

Response LTERM SMITH:

```
GENJCL.IC DBD(HDAMVSAM) DDN(DD1)
//IC135607   JOB
//IC        EXEC PGM=DFSRRC00,PARM='ULU,DFSUDMP0',REGION=800K
//*
//*     THIS JCL ORIGINATES FROM THE USER'S 'JCLPDS' LIBRARY.
//*     %KEYWORDS ARE REPLACED BY THE GENJCL FUNCTION OF
//*     THE IMS DATABASE RECOVERY CONTROL FEATURE.
//*
//*                JCL FOR IMAGE COPY.
//*
//SYSPRINT  DD SYSOUT=A
//RECON1    DD DSN=POCONO1,DISP=SHR
//RECON2    DD DSN=POCONO2,DISP=SHR
//IMS        DD DSN=IMS.DBDLIB,DISP=SHR
//DD1    DD DSN=HDAMVASM,DCB=BUFNO=10,DISP=OLD
//DATAOUT1  DD DSN=IMS.HDAMVSAM.DD1.IC.ICDD1,UNIT=3400,
//             VOL=(PRIVATE,,,1,SER=(******)),
//             LABEL=(1,SL),
//             DISP=(NEW,KEEP),DCB=BUFNO=10
//DFSVSAMP  DD *
1024,2
4096,4
//SYSIN     DD *
D1 HDAMVSAM DD1      DATAOUT1
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMG COMMAND COMPLETED
```

Example for /RMINIT command

Entry ET (with comments):

```
/RMINIT DBRC='DB DBD(DIVNTZ04) SHARELVL(3)'. *COMMENT*.
```

Response ET:

```
INIT.DB DBD(DIVNTZ04) SHARELVL(3)
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMI COMMAND COMPLETED
```

Examples for /RMLIST command

Entry ET (with comments):

```
/RMLIST DBRC='DB DBD(DIVNTZ04)'. *LAST COMMENT*.
```

Response ET for full-function databases:

```
         IMS VERSION 11 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0001
   LIST.DB DBD(DIVNTZ02)
10.313 11:47:46.498180              LISTING OF RECON                  PAGE 0002
--------------------------------------------------------------------------------
 DB
  DBD=DIVNTZ02                 IRLMID=*NULL        DMB#=1       TYPE=IMS
  SHARE LEVEL=3               GSGNAME=**NULL**     USID=0000000001
  AUTHORIZED USID=0000000000  RECEIVE USID=0000000000 HARD USID=0000000000
  RECEIVE NEEDED USID=0000000000
  DBRCVGRP=**NULL**
  FLAGS:                              COUNTERS:
    BACKOUT NEEDED        =OFF          RECOVERY NEEDED COUNT   =0
    READ ONLY             =OFF          IMAGE COPY NEEDED COUNT =0
    PROHIBIT AUTHORIZATION=OFF          AUTHORIZED SUBSYSTEMS   =0
    RECOVERABLE           =YES          HELD AUTHORIZATION STATE=0
                                        EEQE COUNT              =0
    TRACKING SUSPENDED    =NO           RECEIVE REQUIRED COUNT  =0
    OFR REQUIRED          =NO
    REORG INTENT          =NO
    QUIESCE IN PROGRESS   =NO
    QUIESCE HELD          =NO
--------------------------------------------------------------------------------
DSP0180I  NUMBER OF RECORDS LISTED IS        1
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 10.313 11:47:46.763069
         IMS VERSION 11 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0003
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
```

Response ET for Fast Path databases:

```
         IMS VERSION 11 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0001
   LIST.DB DBD(DEDBJN21)
10.313 11:47:47.414528              LISTING OF RECON                  PAGE 0002
--------------------------------------------------------------------------------
 DB
  DBD=DEDBJN21                                      DMB#=4       TYPE=FP
  SHARE LEVEL=3
  FLAGS:                              COUNTERS:
                                        RECOVERY NEEDED COUNT   =0
                                        IMAGE COPY NEEDED COUNT =0
    PROHIBIT AUTHORIZATION=OFF          AUTHORIZED AREAS        =0
    RECOVERABLE           =YES          EEQE COUNT              =0
--------------------------------------------------------------------------------
DSP0180I  NUMBER OF RECORDS LISTED IS        1
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 10.313 11:47:47.663651
         IMS VERSION 11 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0003
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
```

Entry ET:

```
/RMLIST DBRC='DB DBD(DIVNTZ02) DBDS'
```

Response ET for full-function databases:

```
         IMS VERSION 11 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0001
   LIST.DB DBD(DIVNTZ02)  DBDS
10.313 11:47:46.952809              LISTING OF RECON                  PAGE 0002
```

```
--------------------------------------------------------------------------------
 DB
  DBD=DIVNTZ02                IRLMID=*NULL        DMB#=1       TYPE=IMS
  SHARE LEVEL=3              GSGNAME=**NULL**     USID=0000000001
  AUTHORIZED USID=0000000000  RECEIVE USID=0000000000 HARD USID=0000000000
  RECEIVE NEEDED USID=0000000000
  DBRCVGRP=**NULL**
  FLAGS:                              COUNTERS:
    BACKOUT NEEDED        =OFF          RECOVERY NEEDED COUNT   =0
    READ ONLY             =OFF          IMAGE COPY NEEDED COUNT =0
    PROHIBIT AUTHORIZATION=OFF          AUTHORIZED SUBSYSTEMS   =0
    RECOVERABLE           =YES          HELD AUTHORIZATION STATE=0
                                        EEQE COUNT              =0
    TRACKING SUSPENDED    =NO           RECEIVE REQUIRED COUNT  =0
    OFR REQUIRED          =NO
    REORG INTENT          =NO
    QUIESCE IN PROGRESS   =NO
    QUIESCE HELD          =NO
--------------------------------------------------------------------------------
10.313 11:47:46.952809              LISTING OF RECON                  PAGE 0003
--------------------------------------------------------------------------------
 DBDS
  DSN=IMSTESTL.DIVNTZ02.FJXXS01K                                    TYPE=IMS
  DBD=DIVNTZ02  DDN=DBHVSAM1 DSID=001 DBORG=HISAM  DSORG=VSAM
  CAGRP=**NULL**  GENMAX=2     IC AVAIL=0     IC USED=0     DSSN=00000000
  NOREUSE         RECOVPD=0
  DEFLTJCL=**NULL**  ICJCL=ICJCL     OICJCL=OICJCL    RECOVJCL=RECOVJCL
  RECVJCL=ICRCVJCL
  FLAGS:                              COUNTERS:
    IC NEEDED      =OFF
    IC RECOMMENDED =ON
    RECOV NEEDED   =OFF
    RECEIVE NEEDED =OFF                 EEQE COUNT              =0
--------------------------------------------------------------------------------
10.313 11:47:46.952809              LISTING OF RECON                  PAGE 0004
--------------------------------------------------------------------------------
 DBDS
  DSN=IMSTESTL.DIVNTZ02.FJXXS01E                                    TYPE=IMS
  DBD=DIVNTZ02  DDN=DBHVSAM2 DSID=002 DBORG=HISAM  DSORG=VSAM
  CAGRP=**NULL**  GENMAX=2     IC AVAIL=0     IC USED=0     DSSN=00000000
  NOREUSE         RECOVPD=0
  DEFLTJCL=**NULL**  ICJCL=ICJCL     OICJCL=OICJCL    RECOVJCL=RECOVJCL
  RECVJCL=ICRCVJCL
  FLAGS:                              COUNTERS:
    IC NEEDED      =OFF
    IC RECOMMENDED =ON
    RECOV NEEDED   =OFF
    RECEIVE NEEDED =OFF                 EEQE COUNT              =0
--------------------------------------------------------------------------------
DSP0180I  NUMBER OF RECORDS LISTED IS        3
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 10.313 11:47:47.207470
         IMS VERSION 11 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0005
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
```

Response ET for Fast Path databases:

```
         IMS VERSION 11 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0001
   LIST.DB DBD(DEDBJN21)  DBDS
10.313 11:47:47.871802              LISTING OF RECON                  PAGE 0002
--------------------------------------------------------------------------------
 DB
  DBD=DEDBJN21                                      DMB#=4       TYPE=FP
  SHARE LEVEL=3
  FLAGS:                             COUNTERS:
                                       RECOVERY NEEDED COUNT   =0
```

```
                                        IMAGE COPY NEEDED COUNT =0
   PROHIBIT AUTHORIZATION=OFF           AUTHORIZED AREAS        =0
   RECOVERABLE           =YES           EEQE COUNT              =0
--------------------------------------------------------------------------------
10.313 11:47:47.871802             LISTING OF RECON                 PAGE 0003
--------------------------------------------------------------------------------
 DBDS
  DBD=DEDBJN21  AREA=DB21AR0                                    TYPE=FP
  SHARE LEVEL=3             DSID=00001 DBORG=DEDB   DSORG=VSAM
  GSGNAME=**NULL**             USID=0000000002
  AUTHORIZED USID=0000000002  RECEIVE USID=0000000002 HARD USID=0000000002
  RECEIVE NEEDED USID=0000000000
  CAGRP=**NULL**  GENMAX=2     IC AVAIL=0     IC USED=0     DSSN=00000001
  NOREUSE         RECOVPD=0      NOVSO  NOPREOPEN  NOPRELOAD
  CFSTR1=**NULL**         CFSTR2=**NULL**          NOLKASID  NOMAS
  DEFLTJCL=**NULL**  ICJCL=ICJCL    RECVJCL=ICRCVJCL  RECOVJCL=RECOVJCL
  DBRCVGRP=**NULL**
  FLAGS:                              COUNTERS:
    PROHIBIT AUTHORIZATION=OFF          AUTHORIZED SUBSYSTEMS   =0
                                        HELD AUTHORIZATION STATE=0
    IC NEEDED             =OFF          ADS AVAIL #             =1
    IC RECOMMENDED        =ON
    RECOV NEEDED          =OFF          REGISTERED ADS #        =1
                                        EEQE COUNT              =0
    RECEIVE NEEDED        =OFF
    OFR REQUIRED          =NO
    TRACKING SUSPENDED    =NO
    HSSP CIC IN PROGRESS  =NO
    QUIESCE IN PROGRESS   =NO
    QUIESCE HELD          =NO

    ADS LIST:
                                                                    CREATE
    -ADS DDN--ADS DSN-                                        -STAT- -RUNNING-
    DB21AR0  IMSTESTL.DB21AR0                                 AVAIL     NO
--------------------------------------------------------------------------------
 ALLOC
  ALLOC   =10.313 11:30:06.950181         *  ALLOC LRID =0000000000000000
  DSSN=0000000001 USID=0000000002 START = 10.313 11:27:19.900000
10.313 11:47:47.871802             LISTING OF RECON                 PAGE 0004
--------------------------------------------------------------------------------
 DBDS
  DBD=DEDBJN21  AREA=DB21AR1                                    TYPE=FP
  SHARE LEVEL=3             DSID=00002 DBORG=DEDB   DSORG=VSAM
  GSGNAME=**NULL**             USID=0000000002
  AUTHORIZED USID=0000000002  RECEIVE USID=0000000002 HARD USID=0000000002
  RECEIVE NEEDED USID=0000000000
  CAGRP=**NULL**  GENMAX=2     IC AVAIL=0     IC USED=0     DSSN=00000001
  NOREUSE         RECOVPD=0      NOVSO  NOPREOPEN  NOPRELOAD
  CFSTR1=**NULL**         CFSTR2=**NULL**          NOLKASID  NOMAS
  DEFLTJCL=**NULL**  ICJCL=ICJCL    RECVJCL=ICRCVJCL  RECOVJCL=RECOVJCL
  DBRCVGRP=**NULL**
  FLAGS:                              COUNTERS:
    PROHIBIT AUTHORIZATION=OFF          AUTHORIZED SUBSYSTEMS   =0
                                        HELD AUTHORIZATION STATE=0
    IC NEEDED             =OFF          ADS AVAIL #             =1
    IC RECOMMENDED        =ON
    RECOV NEEDED          =OFF          REGISTERED ADS #        =1
                                        EEQE COUNT              =0
    RECEIVE NEEDED        =OFF
    OFR REQUIRED          =NO
    TRACKING SUSPENDED    =NO
    HSSP CIC IN PROGRESS  =NO
    QUIESCE IN PROGRESS   =NO
    QUIESCE HELD          =NO

  ADS LIST:
```

```
                                                                        CREATE
   -ADS DDN--ADS DSN-                                                -STAT- -RUNNING-
   DB21AR1  IMSTESTL.DB21AR1                                         AVAIL     NO
--------------------------------------------------------------------------------
```

Entry ET (With comments):

```
/RML DBRC='DBDS DBD(DEDBJN21) AREA(DB21AR0)'. *VSO AREA*.
```

Response ET:

```
LIST.DBDS DBD(DEDBJN21) AREA(DB21AR0)
------------------------------------------------------------------------
 DBDS
  DBD=DEDBJN21  AREA=DB21AR0                                        TYPE=FP
  SHARE LEVEL=1             DSID=001 DBORG=DEDB   DSORG=VSAM
  GSG NAME=**NULL**            USID=0000000002
  AUTHORIZED USID=0000000002  RECEIVE USID=0000000002  MAX USID=0000000002
  RECEIVE NEEDED USID=0000000000
  CAGRP=**NULL**  GENMAX=5     IC AVAIL=0     IC USED=0     DSSN=0000000
  HSSP IC IN PROCESS=NO          AVAIL         USED         PARTIAL
                                HSSP IC=0     HSSP IC=0     HSSP IC=0
  REUSE           RECOVPD=0      VSO    PREOPEN    PRELOAD
  DEFLTJCL=**NULL**  ICJCL=ICJCL    RECVJCL=ICRCVJCL  RECOVJCL=RECOVJCL
  FLAGS:                            COUNTERS:
    PROHIBIT AUTHORIZATION=OFF        AUTHORIZED SUBSYSTEMS   =1
                                      HELD AUTHORIZATION STATE=7
    IC NEEDED             =OFF        ADS AVAIL #             =1
    RECOV NEEDED          =OFF        REGISTERED ADS #        =1
                                      EEQE COUNT              =0
    TRACKING IN PROGRESS  =NO         RECEIVE NEEDED          =OFF
    OFR REQUIRED          =NO
    TRACKING SUSPENDED    =NO
    HSSP CIC IN PROGRESS  =NO
    QUIESCE IN PROGRESS   =YES
    QUIESCE HELD          =NO
  ADS LIST:
                                                                    CREATE
     -ADS DDN--ADS DSN-                                          -STAT- -RUNNING-
     DB21AR0  DB21AR0                                            AVAIL     NO
   ASSOCIATED SUBSYSTEM INFORMATION:
                               ENCODED
      -SSID-   -ACCESS INTENT-  -STATE-  -SS ROLE-
     SYS3         EXCLUSIVE        7      ACTIVE
  ALLOC
   ALLOC   = 93.076   13:30:35.0*     START   = 93.076   13:30:23.3
   LRID=0000000000000000        DSSN=0000000001     USID=0000000002
  REORG
   RUN     = 93.076   13:23:38.0*     USID=0000000000
 DSP0180I  NUMBER OF RECORDS LISTED IS        3
 DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
 DSP0220I  COMMAND COMPLETION TIME 93.076   13:37:36.7
 DSP0211I  COMMAND PROCESSING COMPLETE
 DSP0211I  HIGHEST CONDITION CODE = 00
 DSP0058I  RML COMMAND COMPLETED
```

Entry ET (With comments):

```
/RML DBRC='DBDS DBD(DEDBJN21) AREA(DB21AR6)'. *DEDB AREA*.
```

Response ET:

```
LIST.DBDS DBD(DEDBJN21) AREA(DB21AR6)
------------------------------------------------------------------------
 DBDS
  DBD=DEDBJN21  AREA=DB21AR6                                        TYPE=FP
  SHARE LEVEL=1             DSID=007 DBORG=DEDB   DSORG=VSAM
  GSG NAME=**NULL**            USID=0000000002
```

```
AUTHORIZED USID=0000000002  RECEIVE USID=0000000002  MAX USID=00000000
RECEIVE NEEDED USID=0000000000
CAGRP=**NULL**  GENMAX=5     IC AVAIL=0     IC USED=0     DSSN=0000000
HSSP IC IN PROCESS=NO          AVAIL         USED         PARTIAL
                              HSSP IC=1     HSSP IC=0     HSSP IC=0
REUSE           RECOVPD=0      NOVSO  NOPREOPEN  NOPRELOAD
DEFLTJCL=**NULL**  ICJCL=ICJCL    RECVJCL=ICRCVJCL  RECOVJCL=RECOVJCL
FLAGS:                            COUNTERS:
  PROHIBIT AUTHORIZATION=OFF        AUTHORIZED SUBSYSTEMS   =0
                                    HELD AUTHORIZATION STATE=0
  IC NEEDED             =OFF        ADS AVAIL #             =0
  RECOV NEEDED          =ON         REGISTERED ADS #        =1
                                    EEQE COUNT              =0
  TRACKING IN PROGRESS  =NO         RECEIVE NEEDED          =OFF
  OFR REQUIRED          =NO
  TRACKING SUSPENDED    =NO
  HSSP CIC IN PROGRESS  =NO
  QUIESCE IN PROGRESS   =YES
  QUIESCE HELD          =NO
ADS LIST:
                                                                CREATE
  -ADS DDN--ADS DSN-                                    -STAT- -RUNNING
  DB21AR6  DB21AR6                                      UNAVAIL   NO
ALLOC
 ALLOC   = 93.076   13:32:05.0*     START   = 93.076   13:30:23.3
 LRID=0000000000000000        DSSN=0000000001     USID=0000000002
 DEALLOC = 93.076   13:34:21.2      DEALLOC LRID=0000000000000000
REORG
 RUN     = 93.076   13:24:12.5*     USID=0000000000
                          AVAILABLE DATA SET
                        DBD=DEDBJN21 DDN=DB21AR6
IMAGE
* CREATE  = 93.076   13:22:52.8*     HSSP
 IC1
  DSN=IC21AR6
DSP0180I  NUMBER OF RECORDS LISTED IS        4
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 93.076   13:38:21.0
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RML COMMAND COMPLETED
```

Example for /RMNOTIFY command

Entry ET (with comments):

```
/RMNOTIFY DBRC='SUBSYS SSID(IMSB) IRLMID(IRLM1) NORMAL'.
```

Response ET:

```
NOTIFY.SUBSYS SSID(IMSB) IRLMID(IRLM1) NORMAL
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE.
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMN COMMAND COMPLETED
```

Related reference:

- CHANGE commands (Commands)
- DELETE commands (Commands)
- GENJCL commands (Commands)
- INIT commands (Commands)
- LIST commands (Commands)

NOTIFY commands (Commands)

Chapter 15. /RSTART command

The /RSTART command starts lines, lines and physical terminals, logical links, nodes, and users when you do not want to reset all associated conditions such as a conversation or special operating mode.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 428
- "Usage notes" on page 430
- "Equivalent IMS type-2 commands" on page 430
- "Examples" on page 430

Environment

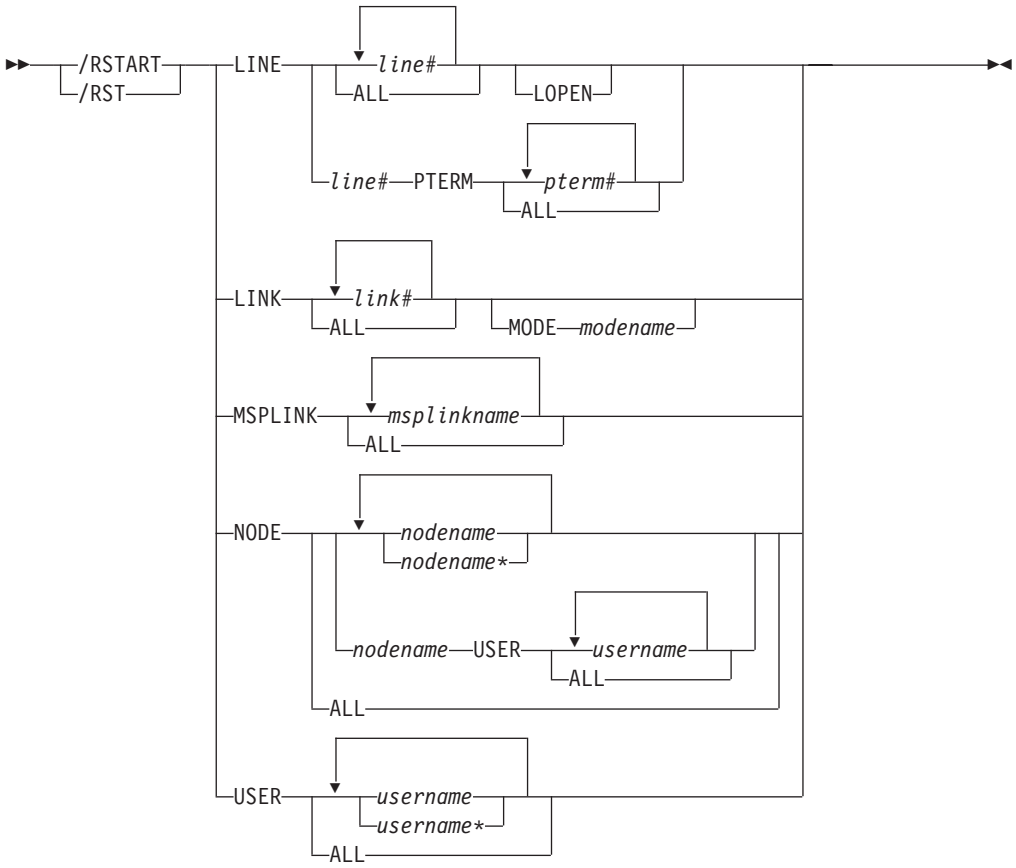
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 188. Valid environments for the /RSTART command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RSTART	X		X
CONTINUOUS	X		X
LINE	X		X
LINK	X		X
LOPEN	X		X
MODE	X		X
MSPLINK	X		X
NODE	X		X
PTERM	X		X
USER	X		X

Syntax

© Copyright IBM Corp. 1974, 2015



Keywords

The following keywords are valid for the /RSTART command:

LINE
Specifies the line or line/physical terminal to be started with all terminals on the line in the same mode of operation they were in when they were stopped. The /RSTART LINE command allows all input, output, and queuing to start on the line and take the line out of response mode if the line was in this mode. The /RSTART LINE PTERM command does not reset line response mode, but does reset the non-VTAM-attached 3270 terminal response mode and looptest mode.

LOPEN
LOPEN enables stopped and idle remote non-VTAM lines. Enter the /RSTART LINE LOPEN command before any /RSTART LINE PTERM command to avoid having a line that is stopped and idle reset before it can be enabled again. If the line is not stopped or process stopped, and idle, or if enabling is not applicable, the LOPEN keyword is ignored and processing continues.

/RSTART LINE and /RSTART NODE cannot reset terminal response mode if Fast Path is active for a specified physical terminal or node. The /DEQUEUE command must be entered to discard Fast Path output before using the /RSTART command.

LINK
Specifies the logical links to be started, either individually or all at once.

For VTAM links, the /RSTART LINK command can be issued in either one of the two partner systems to start communication. For CTC or MTM links, communication does not begin until the /RSTART LINK command is entered in both partner IMS systems.The /RSTART LINK command is rejected unless the link is in a stopped and idled status and the assigned physical link is open, as shown in the /DISPLAY command.

MODE
The MODE keyword allows you to specify mode table entries to be used when activating an IMS VTAM MSC session. Use of the MODE keyword with non-VTAM links is invalid. If non-VTAM links are referred to specifically in the /RSTART LINK command with the MODE keyword, they will be marked in error.

MSPLINK
Resets MSC VTAM links to enable logons.

NODE
Specifies the VTAM node for which input, output, and queuing will start. The /RSTART NODE nodename USER username command restarts the ISC half-session allocated to *username* for *nodename*. The USER keyword is valid only with the NODE keyword and restarts the specified half-session. If the USER keyword is omitted, all half-sessions of the specified node are affected.

Restrictions for using NODE and USER parameters together:

- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- The /RSTART NODE USER commands are valid for ISC, LUP, and 3600 nodes only.

The NODE parameter can be generic if the USER keyword is not specified. The generic parameter specifies nodes that already exist. If the node was created temporarily to retain status data and the status conditions have been reset, then the node is deleted at the next simple checkpoint.

If global resource information is not kept in Resource Manager, the /RSTART NODE command allows a node to logon to the local IMS, without resetting local status. If global resource information is kept in Resource Manager, the /RSTART NODE command allows a node to logon to any IMS in the IMSplex, without resetting global node status kept in Resource Manager. If the node no longer has significant status, it is deleted by Resource Manager.

USER
Specifies the USER for which input, output, and queuing are to start. This command starts the USER without resetting conditions such as conversation mode, exclusive mode, and test mode. The /RSTART USER command applies only to dynamic users.

The USER parameter can be generic where the generic parameter specifies already existing users.

If the user structure is temporary and was created solely to retain status that is now reset, the temporary user is deleted at the next simple checkpoint.

If global resource information is not kept in Resource Manager, the /RSTART USER command allows a user to sign on to the local IMS. If global resource information is kept in Resource Manager, the /RSTART USER command allows

a user to sign on to any IMS in the IMSplex. If the user no longer has significant status in Resource Manager, it is deleted.

Usage notes

The /RSTART command checks the validity of all parameters entered by the terminal operator. If an error is detected on parameters that are independent of one another, only the invalid parameters are flagged as being in error and the /RSTART command processes the rest of the parameters.

The /RSTART command can be used to reset conditions previously established by the /START, /STOP, /PSTOP, /PURGE, /MONITOR, /COMPT, or /RCOMPT command.

This command can be issued to an IMSplex using the Batch SPOC utility.

Equivalent IMS type-2 commands

The following table shows variations of the /RSTART command and the IMS type-2 commands that perform similar functions.

Table 189. Type-2 equivalents for the /RSTART command

Task	/RSTART command	Similar IMS type-2 command
Resets MSC VTAM links to enable logons.	/RSTART MSPLINK *msplinkname* | ALL	UPDATE MSPLINK NAME(*msplinkname* | *) START(LOGON)

Examples

The following are examples of the /RSTART command:

Example 1 for /RSTART command

Entry ET:

```
/RSTART LINE 4,5,6,7,8,9,10,11
```

Response ET:

```
DFS058I  RSTART COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL RSTARTED
```

Explanation: LINES 4,5,6,7,8,9,10, and 11 are started.

Example 2 for /RSTART command

Entry ET:

```
/RSTART LINE 4 5 6 700
```

Response ET:

```
DFS058I  RSTART COMMAND COMPLETED EXCEPT LINE 700
```

Example 3 for /RSTART command

Entry ET:

```
/RSTART LINE 4 PTERM 1, 2
```

Response ET:

```
DFS058I  RSTART COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL RSTARTED
```

Explanation: LINE 4 PTERM 1 and 2 are started.

Example 4 for /RSTART command

Entry ET:

```
/RSTART LINK ALL
```

Response ET:

```
DFS058I  RSTART COMMAND COMPLETED
```

Explanation: All of the logical links are started. For CTC and MTM links, communication across the link does not begin until the partner in the remote system is started with the /RSTART LINK command.

Response ET:

```
DFS2168I  CONNECTION ESTABLISHED ON LINK 2
```

Explanation: The connection for communication between two IMS systems is established. The partner link is started with a /RSTART LINK command. After each connection, the message DFS2168 is returned.

Example 5 for /RSTART command

Entry ET:

```
/RSTART LINK 2
```

Response ET:

```
DFS058I  RSTART COMMAND COMPLETED
```

Explanation: Logical link 2 is started.

Response ET:

```
DFS2168I  CONNECTION ESTABLISHED ON LINK 2
```

Explanation: The two IMS systems are connected.

Example 7 for /RSTART command and a VTAM link

Entry ET:

```
/RSTART LINK 10
```

Response ET:

```
DFS058I 17:34:36 RSTART COMMAND COMPLETED
DFS2168I 17:34:36 CONNECTION ESTABLISHED ON LINK 0010
```

Response ET at the partner system:

```
DFS2160I 17:34:36 LINK 013 STARTED BY PARTNER AK NODE L6APPL3
DFS2168I 17:34:36 CONNECTION ESTABLISHED ON LINK 0013
```

Explanation: In the Response ET at the partner system, L6APPL3 is the APPLID of the partner system in which the /RSTART command was issued.

Example 8 for /RSTART command

Entry ET:

```
/RSTART NODE EAST
```

Response ET:

```
DFS058I  RSTART COMMAND COMPLETED
```

Explanation: The node named EAST is started.

Chapter 16. /RTAKEOVER command

The /RTAKEOVER command requests a remote takeover of an active IMS subsystem by a tracking subsystem at a secondary site.

The remote takeover causes the tracking subsystem to finish processing and shut down.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 435

Environment

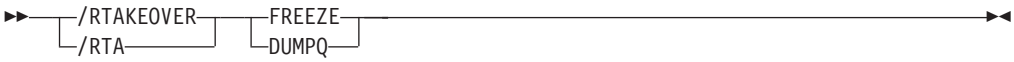
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 190. Valid environments for the /RTAKEOVER command and keywords

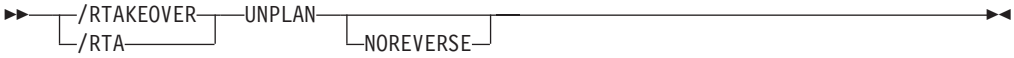
Command / Keywords	DB/DC	DBCTL	DCCTL
/RTAKEOVER	X	X	X
DUMPQ	X	X	X
FREEZE	X	X	X
NOREVERSE	X	X	X
UNPLAN	X	X	X

Syntax

/RTAKEOVER for an active subsystem



/RTAKEOVER for a tracking subsystem



Usage notes

If entered on an active subsystem, /RTAKEOVER also causes the active subsystem to shut down. Once the remote tracking subsystem has shut down and remote takeover has successfully completed, one or more subsystems may be brought up in an active role at the secondary site and started with standard IMS restart commands.

© Copyright IBM Corp. 1974, 2015

Unless you specify the NOREVERSE keyword, a remote takeover can be reversed, even after the /RTAKEOVER command has been issued. The NORTA parameter on the CHANGE.SG command can be used to reverse a remote takeover.

This command can be issued to an IMSplex using the Batch SPOC utility.

/RTAKEOVER for an active subsystem

/RTAKEOVER is issued on the active IMS subsystem to initiate a planned remote takeover and must be specified with either the FREEZE keyword or the DUMPQ keyword.

/RTAKEOVER FREEZE indicates that a /CHECKPOINT FREEZE type of shutdown is performed before the planned takeover occurs.

/RTAKEOVER DUMPQ indicates that a /CHECKPOINT DUMPQ type of shutdown is performed before the planned takeover occurs. This form of takeover provides that all relevant log records reach the tracking subsystem such that no data is lost. This form of takeover allows the capability to rebuild the message queues during the new active start.

/RTAKEOVER must be entered for at least one IMS subsystem in the global service group (GSG) at the active site for which takeover is to occur. The other IMS subsystems at the active site must also be shut down, either with the /RTAKEOVER command, or some form of the /CHECKPOINT command that shuts the system down.

Once the active sends all of its log data sets, the active IMS subsystem shuts down. When all of the active subsystems in the global service group have shut down, the tracking subsystem then completes tracking work, stops online forward recovery (OFR), changes the role of the service group (tracking to active) in the RECON data set, and shuts down.

/RTAKEOVER for a tracking subsystem

/RTAKEOVER UNPLAN is issued on the tracking IMS subsystem to initiate an unplanned remote takeover after the active site fails unexpectedly. /RTAKEOVER UNPLAN causes the tracking subsystem to complete tracking work, stops online forward recovery (OFR), changes the role of the service group (tracking to active) in the RECON data set, and shuts down.

Recommendation: Specify the NOREVERSE keyword to enable the tracking IMS subsystem to save and process all data it has received from the active site, regardless of whether that data was committed on the active IMS subsystem.

Although you can still reverse a remote takeover even if you specify NOREVERSE, you should not reverse it; in this case, you receive message DFS4122A when you restart the tracking subsystem.

If you do not specify NOREVERSE, the tracking IMS subsystem discards any uncommitted data it has received from the active subsystem.

Examples

The following are examples of the /RTAKEOVER command:

Example for /RTAKEOVER command at active site

Entry ET:

```
/RTA FREEZE
```

Response ET:

```
DFS2939I REMOTE SITE PLANNED TAKEOVER IN PROGRESS SYS3
DFS2719I MSDB CHECKPOINT WRITTEN TO MSDBCP2  SYS3 DFS994I
         *CHKPT 94308/160026**FREEZE*  SYS3
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBB  FORMATA  MODSTAT ID:  2 SYS3
DFS3804I LATEST RESTART CHKPT: 94308/160026, LATEST BUILDQ CHKPT: 94308/154950 SYS3
DFS4036I CONVERSATION ENDING WITH SERVICE GROUP STLSITE2 SYS3
DFS3257I ONLINE LOG CLOSED ON DFSOLP01 SYS3
DFS2484I JOBNAME=JT160031 GENERATED BY LOG AUTOMATIC ARCHIVING SYS3
DFS092I  IMS LOG TERMINATED    SYS3
DFS4036I CONVERSATION ENDED  WITH SERVICE GROUP STLSITE2 SYS3
DFS4024I  STOP SERVGRP PROCESSING (INTERNAL) COMPLETE SYS3
DFS2091I IMS TIMER SERVICE SHUTDOWN COMPLETE SYS3
DFS0617I RDS BUFFERS HAVE BEEN SUCCESSFULLY PURGED SYS3
```

Response received at the Tracking system:

```
DFS2932I DATABASE UPDATES PRIOR TO SYSTEM SYS3 TAKEOVER HAVE BEEN
 ROUTED SYS3
```

Example for /RTAKEOVER DUMPQ at active site

Entry ET:

```
/RTA DUMPQ
```

Response ET:

```
DFS2939I REMOTE SITE PLANNED TAKEOVER IN PROGRESS SYS3
DFS2719I MSDB CHECKPOINT WRITTEN TO MSDBCP2  SYS3
DFS994I  *CHKPT 94308/165340**DUMPQ**  SYS3
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBB  FORMATA  MODSTAT ID:  2 SYS3
DFS3804I LATEST RESTART CHKPT: 94308/165340, LATEST BUILDQ CHKPT: 94308/165340 SYS3
DFS4036I CONVERSATION ENDING WITH SERVICE GROUP STLSITE2 SYS3
DFS3257I ONLINE LOG CLOSED ON DFSOLP01 SYS3
DFS2484I JOBNAME=JT165345 GENERATED BY LOG AUTOMATIC ARCHIVING SYS3
DFS092I  IMS LOG TERMINATED    SYS3
DFS4036I CONVERSATION ENDED  WITH SERVICE GROUP STLSITE2 SYS3
DFS4024I  STOP SERVGRP PROCESSING (INTERNAL) COMPLETE SYS3
DFS2091I IMS TIMER SERVICE SHUTDOWN COMPLETE SYS3
DFS0617I RDS BUFFERS HAVE BEEN SUCCESSFULLY PURGED SYS3
```

Response received at the Tracking system:

```
DFS2932I DATABASE UPDATES PRIOR TO SYSTEM SYS3 TAKEOVER HAVE BEEN
ROUTED SYS3
```

Example for /RTAKEOVER UNPLAN at remote site

Entry ET:

```
/RTA UNPLAN
```

Response ET:

```
DFS4123I UNPLANNED TAKEOVER IN PROGRESS
DFS2913I CONVERSATION WITH IMS SYS3 TERMINATED: TRK SYS SHUTDOWN
DFS2913I CONVERSATION WITH IMS IMS2 TERMINATED: TRK SYS SHUTDOWN
DFS2500I DATASET  IMZ00007 SUCCESSFULLY DEALLOCATED
DFS2500I DATASET  IMZ00015 SUCCESSFULLY DEALLOCATED
DFS2934I LOG TRUNCATION STARTED FOR IMS: IMS2
DFS2934I LOG TRUNCATION STARTED FOR IMS: SYS3
DFS2943I THERE ARE NO DATA SETS TO TRUNCATE FOR SYS3
DFS2908I DATABASE UPDATES COMPLETE FOR REMOTE TAKEOVER PROCESSING
DFS2500I DATASET  IMZ00020 SUCCESSFULLY CREATED
DFS2500I DATASET  IMZ00021 SUCCESSFULLY ALLOCATED
DFS2500I DATASET  IMZ00021 SUCCESSFULLY DEALLOCATED
DFS2935I TRACKING LOG DATA SETS HAVE BEEN TRUNCATED AT 000000000000209C:
DFS2936I IMSTESTL.RSR.SLDS1.N0000012
DFS2500I DATASET  IMZ00016 SUCCESSFULLY DEALLOCATED
DFS2500I DATASET  IMZ00020 SUCCESSFULLY DELETED
DFS4126I TAKEOVER COMPLETE
DFS994I *CHKPT 94310/160240**FREEZE*
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:    3
DFS3804I LATEST RESTART CHKPT: 94310/160240, LATEST BUILDQ CHKPT: 94310/155301
DFS3257I ONLINE LOG CLOSED ON DFSOLP01
DFS2484I JOBNAME=JT160245 GENERATED BY LOG AUTOMATIC ARCHIVING
DFS092I IMS LOG TERMINATED
DFS2091I IMS TIMER SERVICE SHUTDOWN COMPLETE
DFS0617I RDS BUFFERS HAVE BEEN SUCCESSFULLY PURGED
```

Explanation: An unplanned takeover is successfully initiated for a tracking subsystem that was tracking 2 active subsystems (SYS3 and IMS2).

Chapter 17. /SECURE command

The /SECURE command is used to control the RACF security level. It is used for administrative control of the IMS environment and as an emergency operations control command to throttle RACF activity without requiring an IMS shutdown.

This command can be issued to an IMSplex using the Batch SPOC utility.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 438
- "Examples" on page 439

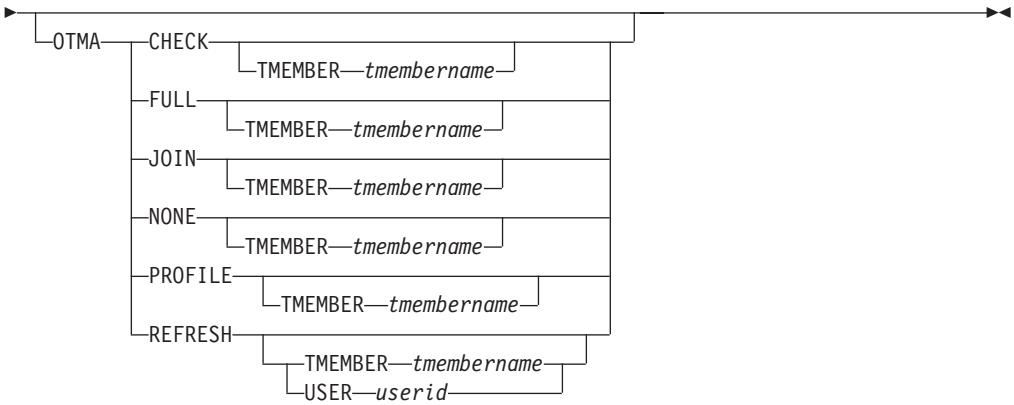
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

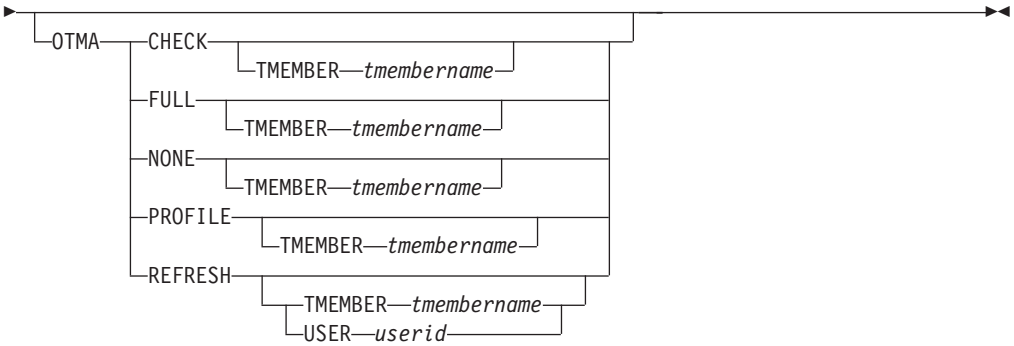
Table 191. Valid environments for the /SECURE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/SECURE	X		X
APPC	X		X
OTMA	X		X

Syntax



© Copyright IBM Corp. 1974, 2015



Keywords

The following keywords are valid for the /SECURE command:

APPC
When used with the CHECK, FULL, NONE, or PROFILE parameters. APPC controls the RACF security level for input from LU 6.2 devices. The /DISPLAY APPC command can be used to show the security level that is currently in effect. At IMS startup, the security default is FULL.

CHECK
Causes existing RACF calls to be made. IMS commands are checked using the RACF resource class of CIMS. IMS transactions are checked using TIMS. Disables z/OS System Authorization Facility security for IMS allocate PSBs (APSBs).

FULL
Causes the same processing as the CHECK parameter but uses additional RACF calls to create the security environment for dependent regions and enables z/OS System Authorization Facility security for IMS APSBs for all CPI Communications driven application programs.

NONE
Does not call RACF within IMS for security verification. RACF security verification in APPC/MVS is not affected. Disables z/OS System Authorization Facility security for IMS APSBs.

PROFILE
Causes the values in the TP profile for each transaction to be used. If the TP profile is not defined for a transaction, or if the TP profile does not specify a RACF security level, then the default security is CHECK.

OTMA
Is used with the CHECK, FULL, NONE, or PROFILE parameters to control the RACF security level for input from IMS Open Transaction Manager Access (OTMA) clients. The /DISPLAY OTMA command can be used to show the security level that is currently in effect. After an IMS cold start, the security default is FULL if the IMS startup parameter OTMASE= is not used.

CHECK TMEMBER *tmembername*
Causes existing RACF calls to be made for input from the specified OTMA client.

FULL TMEMBER *tmembername*
Causes the same processing as the CHECK parameter for input from the specified OTMA client, but uses additional RACF calls to create the security environment for dependent regions.

JOIN TMEMBER *tmembername*
Causes existing RACF calls to be made on only the OTMA client bid requests from OTMA clients. No transaction or command security checking is performed on individual messages.

NONE TMEMBER *tmembername*
Specifies that there is no RACF security checking within IMS for the input from the specified OTMA client.

PROFILE TMEMBER *tmembername*
Specifies that the values in the Security Data section of the OTMA message prefix of each transaction are used to check security for input from the specified OTMA client.

REFRESH
OTMA caches the ACEE for a user ID to reduce the amount of RACF I/O. As a result, a refresh for the cached ACEE is needed after the RACF database is updated. Issuing the /SEC OTMA REFRESH command without the TMEMBER option performs the ACEE refresh for all user IDs for all the OTMA clients. However, the actual ACEE refresh occurs when the next OTMA message for the user ID is received. This is designed to prevent all the RACF ACEE refreshes from happening at one time.

When USER is specified, OTMA refreshes across all TMEMBERs only ACEEs that include the specified user profile.

USER *userid*
An option to refresh only the specified user ID for all OTMA TMEMBERs. *userid* is the 1-8 character name of RACF User Profile to be refreshed.

Examples

The following are examples of the /SECURE command:

Example 1 for /SECURE command

Entry ET:

```
/DIS APPC
```

Response ET:

```
IMSLU     #APPC-CONV SECURITY STATUS   DESIRED
IMSLUNME           0 PROFILE  ENABLED  ENABLED
*91242/163820*
```

Explanation: Enter /DISPLAY APPC to see which security checking option is in effect.

Entry ET:

```
/SECURE APPC FULL
```

Response ET:

```
DFS058I SECURE COMMAND COMPLETED
```

Example 2 for /SECURE command

Entry ET:

```
/DIS OTMA
```

Response ET:

```
GROUP/MEMBER      XCF-STATUS   USER-STATUS    SECURITY
HARRY
-APPL8            ACTIVE       SERVER         FULL
*95068/175730*
```

Explanation: Enter /DISPLAY OTMA to see which security checking option is in effect.

Entry ET:

```
/SECURE OTMA FULL
```

Response ET:

```
DFS058I SECURE COMMAND COMPLETED
```

Related reference:

Parameter descriptions for IMS procedures (System Definition)

Chapter 18. /SET command

The /SET command establishes the destination of all messages entered into this terminal to another terminal or to a particular transaction code.

If the terminal is in conversation, the /SET command also sets the destination of only the next message to the specified transaction.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 442
- "Examples" on page 443

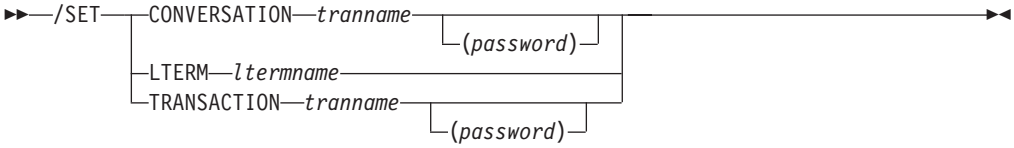
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 192. Valid environments for the /SET command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/SET	X		X
CONVERSATION	X		X
LTERM	X		X
TRANSACTION	X		X

Syntax



Keywords

The following keywords are valid for the /SET command:

CONVERSATION
Directs the next input message to a selected transaction. The terminal being used must be in a conversation that is waiting for an input message. For example, the response message must have been received.

In an IMSplex, if global resource information is kept in Resource Manager, the /SET command sets a transaction for the next input message both globally and locally. If global resource information is not kept in Resource Manager, the /SET command sets the transaction just locally.

© Copyright IBM Corp. 1974, 2015

LTERM
Specifies the logical terminal that is the destination of all messages entered into this terminal.

The mode established by /SET LTERM is called preset mode. If the preset mode is established from a 3270 and user-defined formats are not being used, input message must be entered from a cleared screen. Preset mode can be reset by:

```
/IAM
/STOP LINE PTERM
/STOP NODE command
/STOP USER command
/RESET
/STOP NODE
/STOP LINE
/STOP USER
```

Once a destination is preset, the terminal operator cannot enter the destination (logical terminal name) as the first part of the message.

In a multiple systems configuration, the name of a remote logical terminal can be specified. If the preset destination is to be deleted (/RESET) or changed (/SET), the command must be entered from some other valid component.

TRANSACTION
Specifies the transaction code that is the destination of all messages entered into this terminal. The mode established by /SET TRANSACTION is called preset mode. If the preset mode is established from a 3270 and user-defined formats are not being used, input messages must be entered from a cleared screen. Preset mode can be reset by:

```
/IAM
/START LINE PTERM
/RESET
/STOP NODE
/STOP LINE
/STOP USER
```

Once a destination is preset, the terminal operator cannot enter the destination (transaction code) as the first part of the message. In a multiple systems configuration, the name of a remote transaction can be specified. The terminal cannot be in conversation.

Usage notes

The status fields of /DISPLAY LINE PTERM, /DISPLAY NODE, or /DISPLAY USER indicate if a physical terminal, node, or user is in preset destination mode (PRST), and display the destination transaction code or logical terminal name.

A transaction name can be defined with password protection in SAF for the CONVERSATION and TRANSACTION keywords. If the resource is not defined to SAF, or is defined and is authorized to the user, the command is processed. If the resource is defined to SAF but not authorized for use, the command is rejected with a DFS2469W message.

The password associated with a signed on user, and specified after a command transaction parameter, will be used to perform a reverification check, if the resource is defined to RACF with 'REVERIFY' specified in the APPLDATA field. Passwords can be mixed case or uppercase depending on what is specified on the PSWDC keyword in the DFSPBxxx IMS.PROCLIB member. If the resource passes

the RACF authorization check, and RVFY=Y is specified as an IMS startup parameter, IMS will verify that the password following the parameter is the same as the password entered during signon for the user that entered the command. If 'REVERIFY' is specified for a resource, but a password is not provided, or the wrong password is provided, the command processing for that resource will be rejected.

Examples

The following are examples of the /SET command:

Example 1 for /SET command

Entry ET:

```
/SET CONVERSATION CONVTRAN(password)
```

Response ET:

```
DFS058I  SET COMMAND COMPLETED
```

Explanation: Any message entered from this terminal is sent to conversation CONVTRAN.

Example 2 for /SET command

Entry ET:

```
/SET LTERM CNTRL
```

Response ET:

```
DFS058I  SET COMMAND COMPLETED
```

Explanation: Any message entered from this terminal is sent to LTERM CNTRL.

Example 3 for /SET command

Entry ET:

```
/SET TRANSACTION IMS(password)
```

Response ET:

```
DFS058I  SET COMMAND COMPLETED
```

Explanation: Any message entered from this terminal is sent to transaction IMS.

Chapter 19. /SIGN command

The /SIGN command is used to sign on and sign off at terminals attached to IMS.

This command enables IMS to identify who is using the terminal and to determine if you are authorized to enter the transaction or command.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 448
- "Examples" on page 448

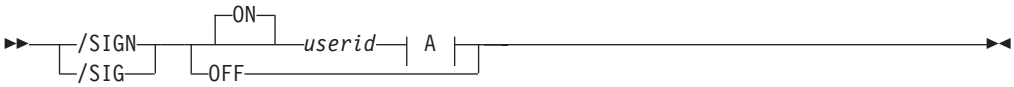
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command can be issued.

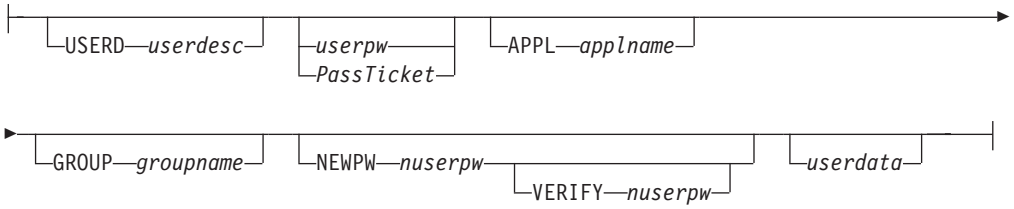
Table 193. Valid environments for the /SIGN command

Command	DB/DC	DBCTL	DCCTL
/SIGN	X		X

Syntax



A:



Keywords

The following keywords are valid for the /SIGN command:

ON /SIGN ON must be issued for any physical terminal or user ID requiring a signon, or the transaction entered is rejected.

From terminals that require signon, commands other than /SIGN or /RCLSDST are rejected if transaction authorization is requested. Static

© Copyright IBM Corp. 1974, 2015

terminals requiring a signon also have enhanced command authorization with RACF or an equivalent product if RCF=S or RCF=A is specified at system startup.

At terminals not requiring signon, transactions are passed to RACF, an equivalent security product, or a user exit for authorization checking. If /SIGN ON is entered at a terminal not requiring a signon, the signon is processed as if the terminal required a signon. That is, the terminal is placed in a signed on status with the user ID until a /SIGN OFF or another /SIGN ON command is entered. For switched terminals, the /IAM command must be issued before the /SIGN ON command.

After any IMS restart or terminal disconnect, the remote terminal operator is required to sign on again using the /SIGN ON command. A terminal can be disconnected by:

- A switched line disconnect
- A VTAM CLSDST
- A line shutdown
- The /IDLE command
- Auto logoff

Signon status is also reset by the /START LINE, /START LINE PTERM, and /START NODE commands and auto signoff.

The remote terminal operator must wait at a static physical terminal for confidential responses, because responses queued for a given physical terminal are sent even if the physical terminal is signed off. If the remote terminal operator must be absent, the /LOCK command can be used to prevent output from being received. Confidential output sent to a dynamic user is queued to the user instead of to the physical terminal when the user has signed off. A successful signon of an existing user turns off the DEADQ status for the user, if that status exists.

For the user exit routine DFSCSGN0, the user ID and userdata parameter values are defined by the installation.

OFF

The /SIGN OFF command is used to complete a session on a terminal that required a signon. Static terminals in conversational mode cannot be signed off without first issuing an /EXIT or /HOLD command.

Another method of signing off a terminal is to reenter the /SIGN ON command. This method initiates a new signon at the terminal without having to enter the /SIGN OFF command.

The /SIGN OFF command resets status that is not significant such as preset mode, test mode, lock lterm, pstop lterm, and purge lterm.

/SIGN OFF for ETO users will also take other actions depending on the recovery settings for the user:

RCVYCONV=NO
: /SIGN OFF causes any IMS conversations (active and held) for an ETO user to be terminated. Any conversational message that is queued or being processed has its output response message delivered asynchronously.

RCVYFP=NO
: /SIGN OFF causes Fast Path status and messages for an ETO user to be discarded.

RCVYRESP=NO
/SIGN OFF resets full-function response mode.

If global resource information is kept in Resource Manager, /SIGN OFF deletes the user ID from Resource Manager (if single user signon enforced) and resets status globally. If the user has no status, /SIGN OFF deletes the user and associated lterms from Resource Manager.

You can specify the following keywords and parameters with the ON keyword:

APPL
A keyword that notifies IMS that the following character string should be the application name used by IMS when IMS makes the SAF call to verify the user. The default application name used by IMS is the IMSID. The IMSID can be overridden by the SAPPLID= parameter in the IMS PROCLIB member DFSDCxxx. If the signon specifies a PassTicket instead of a password, the APPL parameter should specify the application name used when the PassTicket was created. The creator of the PassTicket can specify any value to identify an IMS subsystem.

If RACF is used, APPL= should specify the name of the RACF PTKTDATA profile for IMS as defined to RACF by the creator of the PassTicket. If the name of the PTKTDATA profile is the same as the IMSID, the APPL keyword is not needed.

GROUP
Is an optional keyword indicating a group name of 8 characters or fewer that is associated with the user ID.

NEWPW
Is an optional keyword that indicates a new user password of 8 characters or fewer that replaces the current user password specified in *userpw*. Passwords can be mixed case or uppercase depending on what is specified on the PSWDC keyword in the DFSPBxxx IMS.PROCLIB member.

nuserpw
Is a new password of 8 characters or fewer that is associated with the user identification.

USERD
Is a user descriptor name. This user descriptor name is used in the signon. The *userdesc* parameter must be a user ID, node name, or DFSUSER.

userdata
Is user identification information that has been defined to IMS with the (RACF), equivalent security product or the user exit routine, DFSCSGN0. For RACF, this information consists of the following:

```
userpw  GROUP groupname   NEWPW nuserpw
```

userid
Is a user identification of 8 characters or fewer.

userpw | *PassTicket*
Specifies user identification in one of the following formats:

userpw
Is a password of 8 characters or fewer that is associated with the user identification. Passwords can be mixed case or uppercase depending on what is specified on the PSWDC keyword in the DFSPBxxx IMS.PROCLIB member.

PassTicket
A one-time password that is generated by a requesting product or function. The *PassTicket* is an alternative to the RACF password. Using a *PassTicket* removes the need to send RACF passwords across the network in clear text.

VERIFY
Is an optional keyword that requests IMS to verify the new password entered. IMS verifies the new password before passing it to RACF or to the IMS signon exit routines. This keyword can also be used as an alternative to reentering the password on the DFS3656 panel.

Restriction: You can use this keyword only when responding to an IMS DFS3656A message and as an alternative to reentering the password on the DFS3656 panel.

nuserpw
Is a new password of 8 characters or fewer that is associated with the user identification.

Usage notes

When SGN=G, Z, or M is specified, the user can sign on multiple times to both STATIC and ETO terminals when the structure name is different from the user ID.

For a static terminal, or a dynamic terminal that has the same SPQBname as the node name, a user will not be allowed to sign on unless all conversations are held, or the user is authorized to use the transaction for the active conversation.

If there is an active conversation for a static terminal, and the user is not authorized to use its transaction, the user can enter a /HOLD command prior to signing on to put all of the conversations in a held state. The user will then be allowed to sign on.

If there is an active conversation for a dynamic terminal that has the SPQBname the same as the node name, only a user that is authorized to use the transaction of the active conversation will be allowed to sign on. The /HOLD command is not allowed prior to signing on for a dynamic terminal.

If there is an active conversation for a dynamic terminal that has the SPQBname the same as the USERID, the conversation will be associated with that user at signoff. That same user can sign on to any dynamic terminal and continue the conversation if they are still authorized to use the conversational transaction. Any new user that signs on to the dynamic terminal will not be in a conversation unless they are continuing a conversation from a previous signon or starting a new conversation by entering an authorized conversational transaction.

The status fields of /DISPLAY NODE and /DISPLAY LINE PTERM indicate whether a terminal is signed on with the word SIGN.

Examples

The following are examples of the /SIGN command:

Example 1 for /SIGN ***command***

Entry ET:

```
DFS3649A /SIGN COMMAND REQUIRED FOR IMS

DATE: 11/03/92     TIME: 14:39:33

NODE NAME: DT327001

USERID: IMSUS01

PASSWORD: IMSPW01

USER DESCRIPTOR:
GROUP NAME:
NEW PASSWORD:

   OUTPUT SECURITY AVAILABLE
```

Response ET:

```
DFS3650I SESSION STATUS FOR IMS

DATE: 11/03/92     TIME: 14:41:48
NODE NAME:           DT327001
USERID:              IMSUS01
PRESET DESTINATION:

CURRENT SESSION STATUS:

   OUTPUT SECURITY AVAILABLE
```

Explanation: The user with user ID IMSUS01 and password IMSPW01 has successfully signed on to a dynamic terminal. The signon is done with the panel (DFS3649A).

Example 2 for /SIGN ***command***

Entry ET:

```
/SIGN IMSUS02 IMSPW02
```

Response ET:

```
DFS3650I SESSION STATUS FOR IMS

DATE: 11/03/92     TIME: 14:41:48
NODE NAME:           DT327001
USERID:              IMSUS02
PRESET DESTINATION:

CURRENT SESSION STATUS:

   OUTPUT SECURITY AVAILABLE
```

Explanation: The user with user ID IMSUS02 and password IMSPW02 has successfully signed on to a dynamic terminal. The signon is done with the /SIGN command.

Example 3 for /SIGN ***command***

Entry ET:

```
/SIGN IMSUS03 IMSPW03
```

Response ET:

```
DFS3650I SESSION STATUS FOR IMS

DATE: 11/03/92     TIME: 14:45:53
NODE NAME:           L3270A
USERID:              IMSUS03
PRESET DESTINATION:

CURRENT SESSION STATUS:

NO OUTPUT SECURITY AVAILABLE
```

Explanation: The user with user ID IMSUS03 and password IMSPW03 has successfully signed on to a static terminal.

Related concepts:

z/OS: RACF secured signon PassTicket

Related tasks:

z/OS: Using the secured signon function

Chapter 20. /SMCOPY command

The /SMCOPY command is used by the master terminal operator to control the printing of certain output to the secondary master terminal.

This command can be issued to an IMSplex using the Batch SPOC utility.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 452
- "Example" on page 452

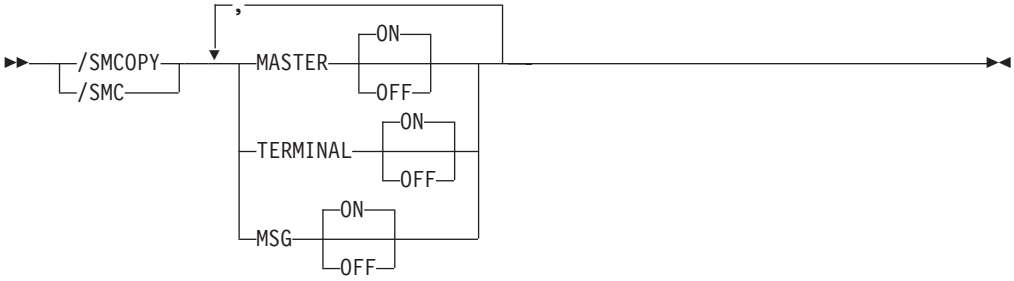
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 194. Valid environments for the /SMCOPY command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/SMCOPY	X		X
MASTER	X		X
MSG	X		X
TERMINAL	X		X

Syntax



Keywords

The following keywords are valid for the /SMCOPY command:

MASTER
Turns ON or OFF the printing of the above subset of IMS commands and command responses when issued from the master terminal.

© Copyright IBM Corp. 1974, 2015

TERMINAL
Turns ON or OFF the printing of the above subset of IMS commands and command responses when issued from terminals other than the master terminal.

Input coming through Operations Manager is not reflected in the secondary master. This also applies to input coming from MCS/E-MCS terminals.

MSG

ON All system messages will be logged to the secondary master. This is the default.

Before logging a message to the secondary master, if the installation type-2 exit routine indicated it needs to be called for messages, the exit routine is called first. The exit routine then determines if the message is logged to the secondary master.

OFF
IMS does not send system messages to the secondary master.

Usage notes

System definition establishes whether the commands and their responses will be printed on the secondary master and the origin of the printed command (master terminal, remote terminal, or both). /SMCOPY provides online control of the printing established by system definition.

Example

A system definition has established that copies of the above subset of commands and command responses, when issued from any terminal, will be printed on the secondary master terminal (operand COPYLOG=ALL was specified on the COMM macro).

Entry ET:

```
/SMCOPY TERMINAL OFF
```

Response ET:

```
DFS058I  SMCOPY COMMAND COMPLETED
```

Explanation: The secondary master terminal does not receive copies of IMS commands and command responses issued from remote terminals. Commands and responses issued from the master terminal are still received.

Related reference:

IMS type-1 commands logged to the secondary master terminal (Commands)

Chapter 21. /SSR command

The /SSR command is a multisegment command that allows the IMS operator to enter an external subsystem command as part of the command input. (The external system is not a CCTL subsystem.)

Subsection:

- "Environment"
- "Syntax"
- "Usage notes"
- "Example"

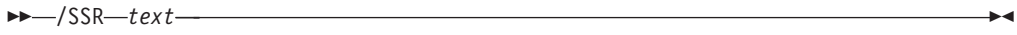
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command can be issued.

Table 195. Valid environments for the /SSR command

Command	DB/DC	DBCTL	DCCTL
/SSR	X	X	X

Syntax



Usage notes

Routing is the only function IMS performs. The command is processed by the external subsystem and the response (from the external subsystem, not CCTL) is routed back to the entering terminal.

All /SSR formats require an EOM indication to denote end-of-message. An EOS indication must be included for all segments that precede the last segment.

text is the alphanumeric external subsystem command.

Example

Entry ET:

```
/SSR ;START DATABASE (DSN8D22P)
```

Response ET:

```
DFS058I  SSR COMMAND COMPLETED

DSN9022I ; DSNTDDIS 'START DATABASE' NORMAL COMPLETION
```

Explanation: The START DATABASE command is successfully routed to the DB2 subsystem for processing.

© Copyright IBM Corp. 1974, 2015

Chapter 22. /START commands

The /START commands make IMS resources available for reference and use.

/START also checks the validity of all parameters entered by the terminal operator. If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and the /START command processes the rest of the parameters. For example,

```
/START LINE 4 6 200
DFS058 START COMMAND COMPLETED EXCEPT LINE 200
```

signifies parameter value 200 is not a valid line number.

When a resource becomes available, the system parameters used for this initialization of IMS are displayed in message DFS1929I. The system parameters are also written to the job log.

These commands can be issued to an IMSplex using the Batch SPOC utility.

/START commands are:

- "/START APPC command" on page 456
- "/START AREA command" on page 456
- "/START AUTOARCH command" on page 460
- "/START CLASS command" on page 461
- "/START DATAGRP command" on page 461
- "/START DB command" on page 465
- "/START DC command" on page 473
- "/START DESC command" on page 474
- "/START ISOLOG command" on page 474
- "/START LINE command" on page 475
- "/START LTERM command" on page 477
- "/START LUNAME command" on page 478
- "/START MADSIOT command" on page 480
- "/START MSNAME command" on page 481
- "/START NODE command" on page 481
- "/START OLDS command" on page 483
- "/START OTMA command" on page 484
- "/START PGM command" on page 485
- "/START REGION command" on page 487
- "/START RTC command" on page 489
- "/START SB command" on page 490
- "/START SERVGRP command" on page 491
- "/START SLDSREAD command" on page 492
- "/START SUBSYS command" on page 493
- "/START SURV command" on page 494
- "/START THREAD command" on page 495
- "/START TMEM command" on page 496
- "/START TRAN command" on page 498
- "/START TRKARCH command" on page 501
- "/START USER command" on page 501
- "/START VGR command" on page 503
- "/START WADS command" on page 504
- "/START XRCTRACK command" on page 505

© Copyright IBM Corp. 1974, 2015

/START APPC command

Use the /START APPC command to instruct IMS to activate the connection to APPC/z/OS and to start accepting transaction schedule requests from APPC/z/OS.

This command reverses the effect of a /PURGE APPC command or a /STOP APPC(CANCEL) command.

The /START APPC command sets the desired status to ENABLED. The current status is initially set to STARTING. When APPC/z/OS responds to the start request, the status changes to either ENABLED or FAILED.

Subsections:

- "Environment"
- "Syntax"

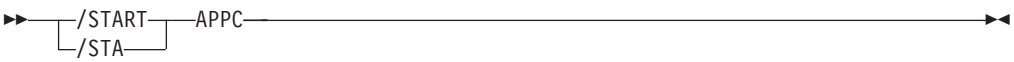
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 196. Valid environments for the /START APPC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
APPC	X		X

Syntax



/START AREA command

Use the /START AREA command to specify the specific areas of DEDBs to be allocated. For z/OS, you can use the /START AREA command to reallocate DEDB areas.

Subsections:

- "Environment"
- "Syntax" on page 457
- "Keywords" on page 457
- "Usage notes" on page 458
- "Equivalent IMS type-2 commands" on page 459
- "Examples" on page 459

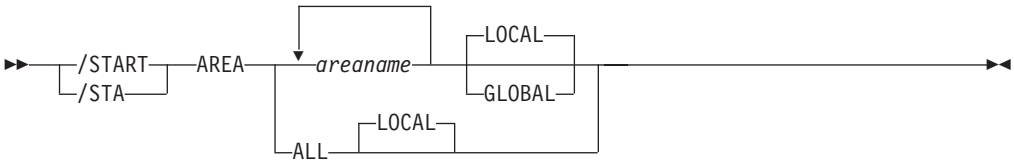
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 197. Valid environments for the /START AREA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
AREA	X		X
GLOBAL	X	X	
LOCAL	X		

Syntax



Keywords

The following keywords are valid for the /START AREA command:

GLOBAL

The GLOBAL keyword applies when an IRLM is active. GLOBAL specifies that the command applies to all subsystems sharing the area. The GLOBAL keyword and the ALL parameter are mutually exclusive. The /START command is rejected if both ALL and GLOBAL are specified. The GLOBAL keyword requires that IRLM be active. The command will be rejected if IRLM is not active.

If the area is registered in the RECON data set, the /STA AREA GLOBAL command will reset the prohibit authorization flag to off (PROHIBIT AUTH = OFF).

The GLOBAL command is processed by the IMS system where the command was initiated. This system will process the command locally and then request IRLM NOTIFY to route and process the command on sharing IMS systems.

If global DB or AREA status is maintained, the global status maintained in RM is also updated. The global status is set to STOA.

If the command is entered from OM API, the global status is updated by the command master IMS. If the command is not entered from OM API, the IMS that initiated the GLOBAL command updates the global status in RM.

You must specify if the global area status must be maintained in RM. You can specify this during IMS initialization in either the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSAREA(Y)). You can also change it dynamically using the UPDATE IMS SET(PLEXPARM(GSTSAREA(Y))) command. If you do not specify that global database status is to be maintained, the GLOBAL keyword is processed as in prior releases, and the global status is not updated.

If global status in RM is successfully updated, message DFS0988I for RSRCTYPE=AREA is issued. If global status is not successfully updated, message DFS3308I is issued, indicating RM failure, and no command response

lines are generated. Any RM error is traced to the OCMD trace table. Users can issue a QRY AREA STATUS(GLOBAL) command to obtain the global status of the resources in RM.

The X'594C' log record for databases is updated to include both global status and global command time stamp.

The GLOBAL keyword is not supported on an RSR tracking subsystem.

If the GLOBAL keyword on a command is entered from an OM API, the command is processed only by the command master IMS. The command master IMS will make DBRC calls to update the RECON with GLOBAL status. It will also request IRLM NOTIFY to route and process the command on sharing IMS systems, and then process the command locally. All other non-master IMS systems ignore the /START command with the GLOBAL keyword.

Messages produced on the NOTIFIED systems will appear only on the system console and will not be routed back to the OM API which originally entered the command.

If multiple IMS systems have been explicitly specified in the route list, the master IMS system will process the command as described previously. However, the non-master IMS systems, to which OM routes the command, will reject the command with the following return and reason code listed in the following table:

Table 198. Return and reason code for GLOBAL keyword issued from the OM API

Return code	Reason code	Meaning
X'00000004'	X'00001000'	The command contained the GLOBAL keyword and was routed to more than one IMS system in the IMSPLEX. The non-master IMS systems will reject this command when OM routes the command to them. The master IMS system will process this command and use IRLM NOTIFY to route and process the command on the non-master IMS systems. See the discussion under the GLOBAL keyword.

LOCAL
Specifies that the command only applies to the IMS subsystem in which the command is entered. This command does not affect any other subsystem sharing the area.

LOCAL is the default.

Usage notes

For areas on an RSR tracking subsystem, /START AREA is used to resume tracking for those areas that were stopped by a previous /DBRECOVERY command or by errors found during tracking subsystem processing. /START AREA also starts online forward recovery (OFR) for those areas that are not current with mainline tracking.

For virtual storage option (VSO) areas that have been defined with the PREOPEN option, /START AREA causes the areas to be preopened. If the VSO area is defined with the PRELOAD option, /START AREA causes the area to be opened and loaded into the z/OS data space.

Restriction: This command only applies to the IMS subsystem on which it is entered; it does not preload or preopen areas on other IMS subsystems in the sysplex that share the area.

The /START AREA command has no effect on VSO areas that are in virtual storage when the command is issued.

The output of the /START AREA command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. For commands that specify GLOBAL, only the command master returns the asynchronous messages to OM. When a command is processed with the LOCAL keyword, all IMS systems are able to return the asynchronous messages to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS0011I, DFS140I, DFS0488I, DFS0666I, DFS1407I, DFS2980E, DFS2981E, DFS3320I, DFS3325I, DFS3342I, DFS3720I, DFS3824I

/START AREA ALL causes message DFS0488 to be issued for every area that is not started successfully, but you do not see a DFS0488 message for every area that does start successfully. You do see a final DFS0488 message which indicates the end of command processing.

While the database is being quiesced, this command cannot be processed successfully.

Equivalent IMS type-2 commands

The following table shows variations of the /START AREA command and the IMS type-2 commands that perform similar functions.

Table 199. Type-2 equivalents for the /START AREA command

Task	/START AREA command	Similar IMS type-2 command
Starts the area.	/START AREA *areaname*	UPDATE AREA NAME(*areaname*) START(ACCESS)

Examples

The following is an example of the /START AREA command:

Entry ET:

```
/START AREA DB1AREA0 DB1AREA1
```

Response ET:

```
DFS058I   START COMMAND IN PROGRESS
DFS0488I  START COMMAND COMPLETED.  AREA=DB1AREA0
DFS0488I  START COMMAND COMPLETED.  AREA=DB1AREA1
```

Explanation: DEDB areas DB1AREA0 and DB1AREA1 are started.

Related concepts:

Maintaining global information for databases, DEDB areas, and transactions (System Administration)

Related reference:

UPDATE AREA command (Commands)

/START AUTOARCH command

Use the /START AUTOARCH command to set the value to change the automatic archiving option selected at system initialization or to set the value to start automatic archiving after a previous /STOP AUTOARCH command.

/START AUTOARCH will not be carried over a warm start if a checkpoint has occurred after the command has been successfully completed, and before a warm start has been issued.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 200. Valid environments for the /START AUTOARCH command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
AUTOARCH	X	X	X

Syntax

Keywords

The following keywords are valid for the /START AUTOARCH command:

#olds

Specifies the number of OLDSs that are to be filled before the /DBRC GENJCL ARCHIVE command is generated. It is optional and defaults to either the value specified at system initialization or to one. If /DBR NOFEOV or /DBD NOFEOV is issued before *nn* OLDSs are filled, the number of OLDSs currently filled will be archived.

Examples

The following is an example of the /START AUTOARCH command:

Entry ET:

```
/START AUTOARCH 4
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: Automatic archiving will be initiated after 4 OLDS data sets are filled.

/START CLASS command

Use the /START CLASS command to specify transaction class, allowing scheduling of application programs to begin.

Message regions must have appropriate classes assigned to them before scheduling will proceed.

Subsections:

- "Environment"
- "Syntax"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 201. Valid environments for the /START CLASS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
CLASS	X		X

Syntax

Examples

The following is an example of the /START CLASS command:

Entry ET:

```
/START CLASS ALL
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: All classes of transactions are made available for scheduling into message processing regions.

/START DATAGRP command

Use the /START DATAGRP command to specify groups of DL/I databases and Fast Path DEDBs to be allocated.

Data groups are logical groupings of databases and areas; they enable simplified command processing for databases. You define a data group in the RECON data set by using the INIT.DBDSGRP command with parameters GRPNAME and DBGRP. DATAGRP is valid on active and RSR tracking subsystems.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 464
- "Equivalent IMS type-2 commands" on page 464

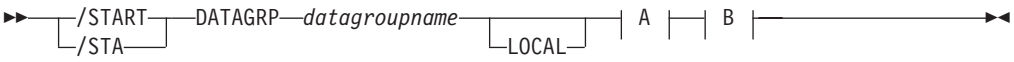
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 202. Valid environments for the /START DATAGRP command and keywords

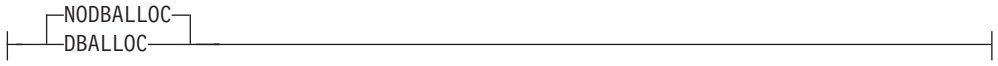
Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
ACCESS	X	X	
DATAGRP	X	X	
LOCAL	X	X	

Syntax



A:

B:



Keywords

The following keywords are valid for the /START DATAGRP command:

ACCESS
Overrides the access intent for a database. The original database access is specified during IMS system definition.

Changing the access intent for a DEDB is allowed only when no PSBs are scheduled that access any areas in the DEDB. You might have to stop PSBs and regions that have wait-for-input (WFI) transactions scheduled before you can change a DEDB's access intent.

Changing the access intent of a database with the /START DB ACCESS= command causes any VSO areas of the database to be removed from virtual storage. Because the access intent of a DEDB cannot be changed while any of its areas are authorized to an IMS subsystem, IMS closes any open areas before processing the command. If a VSO area is closed as a result of the command, it is also removed from the data space.

The /START AREA command must be used to reactivate the VSO options (VSO and PREOPEN) and PRELOAD option for the area. If an area is opened as a result of an access request rather than by the /START AREA command, it is opened as a non-VSO area.

Non-VSO areas with the PREOPEN option are closed as a result of the /START DB ACCESS= command. These areas are reopened either at the next access request for the area or by the /START AREA command.

The GLOBAL and ACCESS keywords are mutually exclusive. The /START command is rejected if both keywords are specified.

The meanings of the ACCESS parameter values are:

RO Specifies that the named database is available for read-only processing on this IMS subsystem.

The only programs which can use the database on this subsystem are those which have a PCB processing option of GO (`PROCOPT=GO`). Programs which access the data using the GO processing option might see uncommitted data, since a sharing IMS subsystem could be updating the database, which is opened for input only.

RD Specifies that the named database is available for read-only processing on this IMS subsystem.

Programs with update intent can be scheduled, but cannot update the database. ACCESS=RD differs from ACCESS=RO in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UP Specifies that the named database is for update as well as read processing in the IMS subsystem.

EX Specifies that the named database is to be used exclusively by this IMS subsystem.

This exclusive access is guaranteed only when the database is registered to DBRC.

NODBALLOC | DBALLOC
Indicates whether the databases within the data group are to be allocated. NODBALLOC is the default.

NODBALLOC
Indicates that the databases within the data group are not to be allocated. The databases will be allocated when they are scheduled. This command does not affect any other subsystem sharing the database.

DBALLOC
Indicates that the databases within the data group are to be allocated.

LOCAL
Specifies that the /START command applies only to the IMS subsystem in which the command is entered.

Usage notes

For databases and areas on an RSR tracking subsystem, /START DATAGRP is used to resume tracking for those areas that were stopped by a previous /DBRECOVERY command or by errors found during tracking subsystem processing. /START DATAGRP also starts online forward recovery (OFR) for those databases and areas that are not current with mainline tracking.

After processing for a /START DATAGRP completes, a DFS0488 message is issued indicating the end of processing. A DFS0488 message is also issued for every database or area that does not start successfully.

If the data group contains both full function and Fast Path databases, a DFS0488 message might be issued indicating the /START DATAGRP command completed successfully before any messages are issued that indicate a Fast Path area did not start successfully. This situation is caused by the asynchronous processing of Fast Path databases.

For virtual storage option (VSO) areas that have been defined with the PREOPEN option, /START AREA causes the areas to be preopened. If the VSO area is defined with the PRELOAD option, /START AREA causes the area to be opened and loaded into the z/OS data space.

A data group is defined in the RECON data set using the INIT.DBDSGRP command with the parameters GRPNAME and DBGRP or DBDSGRP. The DATAGRP keyword on the /START command can specify either a DBDS group or a database group name.

If the ACCESS keyword is specified on the /START DATAGRP command along with the DBDS group name, the ACCESS keyword is not applied to the Fast Path DEDB databases associated with the Fast Path DEDB areas in the DBDS group. If the intent is to use the ACCESS keyword for Fast Path DEDB databases, the DATAGRP parameter must be a database group name that does not contain area names.

Recommendation: Although you can use DBDS groups as well as database groups for this command, you should use database groups whenever possible to eliminate the overhead of converting the DBDS group to a database group.

Equivalent IMS type-2 commands

The following table shows variations of the /START DATAGRP command and the IMS type-2 commands that perform similar functions.

Table 203. Type-2 equivalents for the /START DATAGRP command

Task	/START DATAGRP command	Similar IMS type-2 command
Starts the data group.	/START DATAGRP *datagrpname*	UPDATE DATAGRP NAME(*datagrpname*) START(ACCESS)

Related reference:
"UPDATE DATAGRP command" on page 643

/START DB command

Use the /START DB command to specify the DBD name. The /START DB command permits access from transactions that read or update databases.

The /START DB command can be used to allocate or reallocate all databases other than DEDBs. The /START AREA command must be entered to allocate or deallocate DEDB areas.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 466
- "Usage notes" on page 469
- "Equivalent IMS type-2 commands" on page 471
- "Examples" on page 471

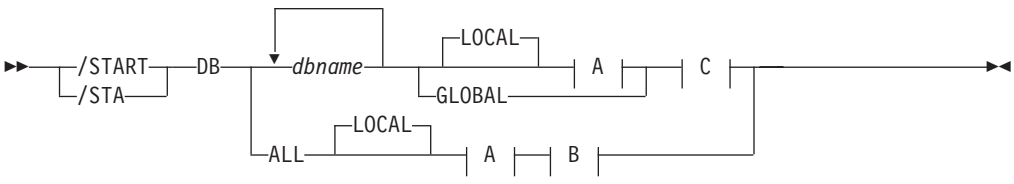
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 204. Valid environments for the /START DB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
ACCESS	X	X	
DB	X	X	
DBALLOC	X	X	
GLOBAL	X	X	
LOCAL	X	X	
NOBACKOUT	X	X	
NODBALLOC	X	X	
NOOPEN	X	X	
OPEN	X	X	

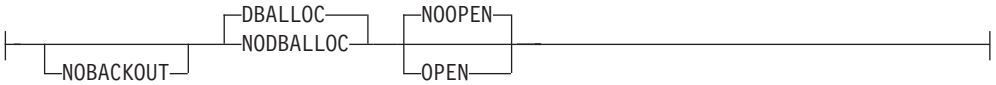
Syntax



A:

B:

C:



Keywords

The following keywords are valid for the /START DB command:

ACCESS
Specifies the type of access that is intended for the named database. This keyword overrides the database access that is specified at system definition. Changing the ACCESS parameter of a DEDB is allowed only when all the AREAS in the DEDB are not authorized by the subsystem.

Changing the access intent of a database with the /START DB ACCESS= command causes any VSO areas of the database to be removed from virtual storage. Because the access intent of a DEDB cannot be changed while any of its areas are authorized to an IMS subsystem, IMS closes any open areas before it processes the command. If a VSO area is closed as a result of the command, it is also removed from the data space.

The /START AREA command must be used to reactivate the VSO options (VSO and PREOPEN) and PRELOAD option for the area. If an area is opened as a result of an access request rather than by the /START AREA command, it is opened as a non-VSO area.

Non-VSO areas with the PREOPEN option are closed as a result of the /START DB ACCESS= command. These areas are reopened either at the next access request for the area or by the /START AREA command.

The GLOBAL and ACCESS keywords are mutually exclusive. The /START command is rejected if both keywords are specified.

Issuing a /START DB command with the ACCESS parameter is not valid for an MSBD.

The meanings of the ACCESS parameter values are:

RO Specifies that the named database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data by using the GO

processing option might see uncommitted data, because a sharing IMS subsystem could be updating the database, which is opened for input only.

RD Specifies that the named database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. ACCESS of RD differs from ACCESS of RO in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UP Specifies that the named database is for update and read processing in the IMS subsystem.

EX Specifies that the named database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC.

DBALLOC
Indicates that the specified databases are to be allocated. DBALLOC is the default except for /START DB ALL commands.

NODBALLOC
Indicates that the specified databases are not to be allocated. The databases will be allocated when they are scheduled. NODBALLOC is the default for /START DB ALL commands.

GLOBAL
Specifies that the command applies to all subsystems that are sharing the database. GLOBAL requires that IRLM be active. The command is rejected if IRLM is not active.

If the database is registered in the RECON data set, the /START DB GLOBAL command will reset the prohibit authorization flag to off (PROHIBIT AUTH = OFF).

The /START DB GLOBAL command is processed by the IMS system where the command was initiated. This system will process the command locally and then request IRLM NOTIFY to route and process the command on sharing IMS systems.

If global DB status is maintained, the global status that is maintained in RM is also updated. The global status is set to STA.

If the command is entered from OM API, the global status is updated by the command master IMS. If the command is not entered from OM API, the IMS that initiated the GLOBAL command updates the global status in RM.

You must specify whether the global database status must be maintained in RM. You can specify this during IMS initialization in the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSDB(Y)). You can also change it dynamically using the UPD IMS SET(PLEXPARM(GSTSDB(Y)) command. If you do not specify that global database status is to be maintained, the GLOBAL keyword is processed as in prior releases, and the global status is not updated.

If global status in RM is successfully updated, message DFS0988I for RSRCTYPE=DB is issued. If global status is not successfully updated, message DFS3308I is issued, indicating RM failure, and no command response lines are generated. Any RM error is traced to the OCMD trace table. Users can issue a QRY DB STATUS(GLOBAL) command to obtain the global status of the resources in RM.

The X'4C' log record for databases is updated to include both global status and global command time stamp. It includes the DEDB name for the area.The GLOBAL keyword and the ALL parameter are mutually exclusive. The /START command is rejected if both ALL and GLOBAL are specified.

The GLOBAL keyword is not supported on an RSR tracking subsystem.

If the GLOBAL keyword on a command is entered from an OM API, the command should only be routed to one IMS system in the IMSplex. The IMS that receives the command from OM will make DBRC calls to update the RECON with GLOBAL status. It will also request IRLM NOTIFY to route and process the command on sharing IMS systems, and then process the command locally.

Messages that are produced on the NOTIFIED systems will be displayed only on the system console and will not be routed back to the OM API that originally entered the command.

If multiple IMS systems have been explicitly specified in the route list, the master IMS system will process the command as described previously. However, the non-master IMS systems, to which OM routes the command, will reject the command with the following return and reason code listed in the following table:

Table 205. Return and reason code for GLOBAL keyword issued from the OM API

Return code	Reason code	Meaning
X'00000004'	X'00001000'	The command contained the GLOBAL keyword and was routed to more than one IMS system in the IMSplex. The non-master IMS systems will reject this command when OM routes the command to them. The master IMS system will process this command and use IRLM NOTIFY to route and process the command on the non-master IMS systems. See the discussion under the GLOBAL keyword.

LOCAL
Specifies that the command applies only to the IMS subsystem in which the command is entered. This command does not affect any other subsystem sharing the database.

LOCAL is the default.

NOBACKOUT
Suppresses backout restart for a database not registered in DBRC. If there was a prior dynamic backout or emergency restart backout failure, then a /START, command will attempt to perform the backout again. However, if the log data required to perform the backout has been archived, the backout must be performed by executing the Batch Backout utility.

If the database is registered in DBRC and is using share control, then DBRC is informed when batch backout is successfully executed, and the failing backout will not be attempted again when the /START command is issued.

If the database is registered in DBRC and is using recovery control, DBRC is not informed when batch backout is successfully executed. You must specify the NOBACKOUT keyword to inform IMS that it does not have to attempt to execute the failed backout again.

NOBACKOUT is not valid with the ALL parameter.

OPEN | NOOPEN

Indicates that the named databases should be opened or should not be opened as part of the /START DB processing. NOOPEN is the default except when the database has EEQEs or the database was previously authorized but not allocated. If the database has EEQEs or was previously authorized but not allocated, then the database will be opened until the NOOPEN keyword is specified. Operators need to be aware of the results of issuing this command. Check the return code in message DFS0488I.

Restriction: The OPEN parameter is not supported:

- On a HALDB master. The command will result in message DFS0488I RC=57, but if only one HALDB master database was specified in the command, the online partition structures of the HALDB master database will be rebuilt if needed. No rebuild will be attempted if more than one database name is listed in the command.
- In an RSR environment.
- On an XRF alternate.
- With keywords ALL, GLOBAL, or NODBALLOC.

Usage notes

For a DEDB, the /START DB command also causes any unloaded randomizer, which is specified in the DBD source, to be reloaded.

You can use the UPD DB START(ACCESS) AREA(*) command to allocate the areas of a DEDB.

When you issue the CREATE DB command for a DEDB, you must specify RESIDENT(Y). DEDBs are always resident. The residency default for the CREATE DB command is non-resident.

After a DEDB is created, you must start it before you can use it in a similar manner as when you add a DEDB or change a DEDB using the online change (OLC) process. After an OLC process that either adds a DEDB or changes a DEDB, the DEDB and the areas must be started. The /START DB command or the UPD DB command will start the database and load the randomizer. The /START AREA command or the UPD AREA command will start the individual areas.

When the name that is specified is for a partition, the action that is taken to allocate data sets varies. The action varies depending on the status of the master database and whether the DMB for the master database is already loaded. If a partition has the /DBRECOVERY command called against it, then the partition cannot be allocated by the /START command even if the DBALLOC keyword is specified. The partition can be allocated by the /START command if the OPEN keyword is used or if the database has EEQEs. The partition will get allocated at first reference if the partition cannot be allocated by the /START command, the OPEN keyword is not used, and the database does not have EEQEs. This applies to partitioned PHDAM and PHIDAM database types.

PHDAM or PHIDAM partitions that had the /DBRECOVERY command issued against them, cannot be allocated with the /START DB DBALLOC command. However, if the partition databases have EEQEs, or the OPEN keyword is used with the /START DB command, or the databases were previously authorized but not allocated, the partitions can be allocated. If you do not use the OPEN keyword, the PSINDEX partition is allocated and the PHDAM and PHIDAM partitions are not allocated until they are authorized.

When a /START DB command is issued for all transactions whose processing program has access to a successfully started database, the USTOPPED attribute will be reset and any messages on the suspend queue for that transaction will be transferred to the normal queue.

If one or more of the named databases requires backout or recovery, and the database is registered in DBRC, the database that require backout or recovery is dropped from the command and the remainder of the databases continue processing. If the database is not registered in DBRC, specify the NOBACKOUT keyword to inform IMS that it does not have to attempt to execute the failed backout again.

To start a HIDAM database, both the index and the data area DBD names must be specified. If a backout failure occurred for this database, the /START command causes the backout to be attempted again.

If the database specified in the command is being used by a batch message processing region, an error message is returned to the master terminal, and the command is ignored for the database that is named in the message. Processing continues for the other databases that are specified in the command. The master terminal operator must wait until the batch message processing concludes before reentering the command.

For databases on an RSR tracking subsystem, /START DB is used to resume tracking for those databases that were stopped by a tracking subsystem processing. The /START DB command also starts online forward recovery (OFR) for those databases that are not current with mainline tracking.

The /START DB ALL command causes message DFS0488I to be issued. All databases that were defined during the system definition process will be started if possible.

The output of the /START DB command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate to the database type and the command completion.

- Full-function database messages: DFS030I, DFS132, DFS160, DFS216, DFS0402I, DFS0488I, DFS0740I, DFS1407, DFS2026, DFS3317I, DFS3318I, DFS3320I, DFS3325I, DFS3465I, DFS3466I
- Fast Path database messages: DFS140I, DFS666, DFS3062

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB except for the /START DB command and the UPDATE DB START(ACCESS) command. A HALDB partition is not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

The HALDB partition reflects conditions such as STOPPED (shown as STOACC or STOSCHD or STOUPDS on QUERY DB), LOCKED, or NOTOPEN. When a HALDB partition is stopped, it must be explicitly started again. The type-1 commands with the keyword ALL, type-2 commands with NAME(*), and

commands against a HALDB master do not change the STOPPED (shown as STOACC, STOSCHD or STOUPDS on QUERY DB) and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is /DBR on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. On a /DISPLAY DB command, only the HALDB master displays STOPPED (each HALDB partition does not display STOPPED unless it was itself stopped). If a /DBR command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

Each partition inherits the access limitations of its HALDB master. If the /DBD command is issued against a HALDB master, all of its partitions close. A subsequent reference to any of the partitions results in the partition opening for input, although the partition's access might be UPDATE or EXCLUSIVE. The DBRC authorization state reflects the limited access.

Restrictions:

- The /START DB command cannot be processed against a HALDB partition on an IMS system while HALDB Online Reorganization (OLR) is running against that partition on the same IMS system.
- The /START DB ACCESS=UP command cannot be issued against a HALDB master while OLR is reorganizing any of its partitions.
- While the database is being quiesced, this command cannot be processed successfully.

Equivalent IMS type-2 commands

The following table shows variations of the /START DB command and the IMS type-2 commands that perform similar functions.

Table 206. Type-2 equivalents for the /START DB command

Task	/START DB command	Similar IMS type-2 command
Starts a database and change access intent of the database.	/START DB ACCESS	UPDATE DB START(ACCESS) SET(ACCTYPE())
Starts a database.	/START DB *dbname*	UPDATE DB NAME(*dbname*) START(ACCESS)

Examples

The following are examples of the /START DB command:

Example 1

TSO SPOC input:

```
STA DB BANKATMS BANKTERM BANKLDGR BE3ORDER
```

TSO SPOC output:

```
SYS3     DFS0488I  STA COMMAND COMPLETED. DBN= BANKATMS RC=04
SYS3     DFS0488I  STA COMMAND COMPLETED. DBN= BANKTERM RC=04
SYS3     DFS0488I  STA COMMAND COMPLETED. DBN= BANKLDGR RC=04
SYS3     DFS0488I  STA COMMAND COMPLETED. DBN= BE3ORDER RC=08
```

```
IMS3     DFS0488I  STA COMMAND COMPLETED. DBN= BANKATMS RC=04
IMS3     DFS0488I  STA COMMAND COMPLETED. DBN= BANKTERM RC=04
IMS3     DFS0488I  STA COMMAND COMPLETED. DBN= BANKLDGR RC=04
IMS3     DFS0488I  STA COMMAND COMPLETED. DBN= BE3ORDER RC=08
```

OM API input:

```
CMD (STA DB BANKATMS BANKTERM BANKLDGR BE3ORDER )
```

OM API output:

```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.197 21:59:29.210362</statime>
<stotime>2002.197 21:59:30.213238</stotime>
<staseq>B7EFC01B367FAE02</staseq>
<stoseq>B7EFC01C2B576D8F</stoseq>
<rqsttkn1>USRT005 10145929</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
</ctl>
<cmderr>
<mbr name="SYS3    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000014</rc>
<rsn>00005050</rsn>
<rsntext>Command processing error</rsntext>
</mbr>
<mbr name="IMS3    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000014</rc>
<rsn>00005050</rsn>
<rsntext>Command processing error</rsntext>
</mbr>
</cmderr>
<cmd>
<master>SYS3    </master>
<userid>USRT005 </userid>
<verb>STA </verb>
<kwd>DB              </kwd>
<input>/STA DB BANKATMS BANKTERM BANKLDGR BE3ORDER   </input>
</cmd>
<msgdata>
<mbr name="SYS3    ">
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKATMS RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKTERM RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKLDGR RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BE3ORDER RC=08</msg>
</mbr>
<mbr name="IMS3    ">
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKATMS RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKTERM RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKLDGR RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BE3ORDER RC=08</msg>
</mbr>
</msgdata>
</imsout>
```

Explanation: The START DB command is routed from OM to the two active IMS systems - SYS3 and IMS3. The response from both IMS systems is returned to OM. The databases BANKATMS, BANKTERM, BANKLDGR, and BE3ORDER are started at both IMS systems.

Example 2

Entry ET:

```
/START DB TREEFARM
```

Response ET:

```
DFS058I  (time stamp) START COMMAND IN PROGRESS
DFS0488I START COMMAND COMPLETED. DBN=TREEFARM RC=0.
```

Explanation: Database TREEFARM is started.

Related concepts:

Maintaining global information for databases, DEDB areas, and transactions (System Administration)

Related reference:

"UPDATE DB command" on page 660

/START DC command

Use the /START DC command to open the VTAM ACBs (if MNPS for XRF is used, then both the MNPS and APPLID ACBs are opened) if they are not already open, to enable logons to IMS, and to enable optional transaction manager functions such as IMS generic resource support and IMS persistent sessions support for RNR.

- IMS generic resource support: The defined VTAM generic resource group is joined with GRSNAME in the IMS or DCC PROCLIB members.
- IMS persistent sessions support for RNR: Session activity that was suspended because of a major outage is resumed or terminated, as appropriate, if the RNR option was specified in the DFSDCxxx PROCLIB member.

If the /START DC command is issued on an XRF alternate system that is using MNPS, the command will only open the APPLID ACB. The MNPS ACB is not opened until XRF takeover processing.

Subsections:

- "Environment"
- "Syntax" on page 474

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 207. Valid environments for the /START DC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
DC	X		X

Syntax

/START DESC command

Use the /START DESC command to define the LU62 descriptors from DFS62xxx PROCLIB member to IMS.

The full PROCLIB member name needs to be specified. The PROCLIB member must start with DFS62.

Subsections:

- "Environment"
- "Syntax"

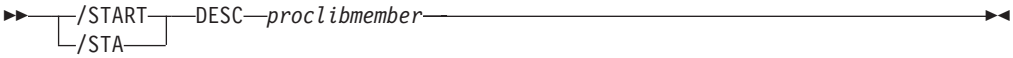
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 208. Valid environments for the /START DESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
DESC	X		X

Syntax



/START ISOLOG command

The /START ISOLOG command indicates that the RSR tracking subsystem is to initiate a request for isolated log data from the isolated log sender at the active site.

/START ISOLOG can only be entered on an RSR tracking subsystem.

/START ISOLOG may be needed when:

- The active system failed and was not restarted, and the OLDS at the time of failure was closed using the Log Recovery utility.
- Batch jobs or utilities running at the active were unable to send logs.
- Permanent error status is shown for gaps in the output of a /DISPLAY TRACKING STATUS command. If the log problem at the active has been corrected, use /START ISOLOG to initiate retry.

Successful completion of syntax checking of the /START ISOLOG command results in the `DFS058 START COMMAND COMPLETED` message, although processing of the command continues asynchronously.

Subsections:

- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 209. Valid environments for the /START ISOLOG command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
ISOLOG	X	X	X

Syntax

/START LINE command

The /START LINE command makes communication lines that are idle and in a stopped or process stopped state available for use. It also terminates any conversations that are active on the line.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes" on page 476
- "Examples" on page 476

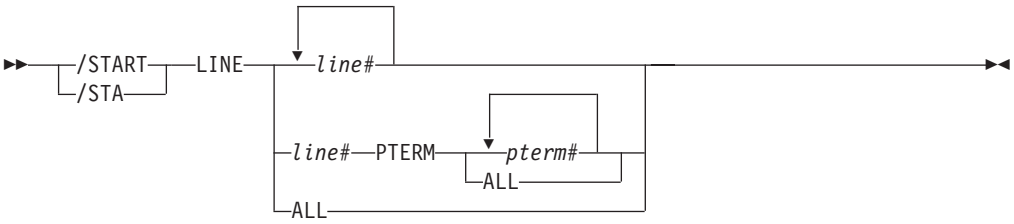
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 210. Valid environments for the /START LINE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
LINE	X		X
PTERM	X		X

Syntax



Usage notes

All terminals are removed from looptest mode, MFSTEST mode, or exclusive mode, and any terminals in signon status are reset. The /START LINE command without the PTERM keyword enables the line again (resulting in a non-VTAM LOPEN macro).

If the line must be enabled again, /START LINE must be entered prior to any /START LINE PTERM command.

The /START LINE PTERM command makes one or more physical terminals available for use. The physical terminals are removed from response mode, test mode, looptest mode, MFSTEST mode, preset destination mode, or exclusive mode, and resets terminals in signon status. If IMS encounters a stopped and idle line when processing /START LINE PTERM, it restarts the line. Any inoperable components are marked as operable.

To activate I/O operations for a LINE, LINE PTERM, or NODE without altering the status of the associated/specified terminals, use /RSTART instead of /START.

If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and processing continues for the rest of the parameters. This happens for /START LINE if:

- The specified line is already started or is not idle.
- Any terminals on the line had conversations that could not be canceled (because an application program is scheduled).
- The specified line could not be started because of IMS internal processing.
- The DD statement is missing from the IMS execution JCL.

The /START LINE command no longer resets preset mode, test mode, and response mode since these statuses are no longer significant and therefore are not kept after a /START LINE command or restart.

Examples

The following are examples of the /START LINE command:

Example 1 for /START LINE command

Entry ET:

```
/START LINE 4 PTERM 1, 2
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL STARTED
```

Explanation: Physical terminals 1 and 2 on line 4 are started.

Example 2 for /START LINE command

Entry ET:

```
/START LINE 4,5,6,7,8,9,10,11
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL STARTED
```

Explanation: Lines 4, 5, 6, 7, 8, 9, 10, and 11 are started.

Example 3 for /START LINE command

Entry ET:

```
/START LINE 4 5 6 700
```

Response ET:

```
DFS058I  START COMMAND COMPLETED EXCEPT LINE 5 700
```

Response RT:

```
DFS059I  TERMINAL STARTED
```

Explanation: Lines 4 and 6 are started. The /DISPLAY LINE command can be used to determine why line 5 did not start successfully. (700 is an invalid line number.)

/START LTERM command

Use the /START LTERM command to specify the logical terminals to be started and to reset the QLOCK state. (QLOCK indicates that the LTERM is locked from sending any further output or from receiving input that can create additional output for the same LTERM until the state is reset by a specific request received on the session.) /START LTERM is rejected for remote logical terminals.

Subsections:

- "Environment"
- "Syntax" on page 478
- "Usage notes" on page 478

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

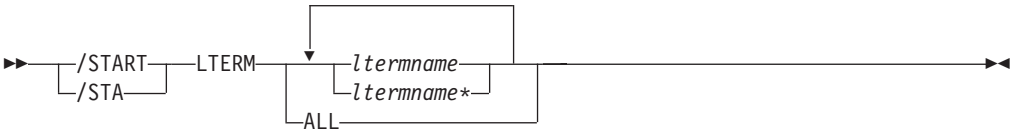
Table 211. Valid environments for the /START LTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X

Table 211. Valid environments for the /START LTERM command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
LTERM	X		X

Syntax



Usage notes

The LTERM keyword is only effective for existing LTERMs.

The LTERM parameter can be generic where the generic parameter specifies LTERMs that already exist.

If global resource information is kept in Resource Manager, the/START LTERM command allows messages to be queued to the LTERM from anywhere in the IMSplex and the change is reflected both in Resource Manager and in the local IMS system.

Examples

The following is an example of the /START LTERM command:

Entry ET:

```
/START LTERM APPLE, TREE, FRUIT
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL STARTED
```

Explanation: Logical terminals APPLE, TREE, and FRUIT are started.

/START LUNAME command

Use the START LUNAME command to specify the LU name that is to be started.

Subsections:

- "Environment" on page 479
- "Syntax" on page 479
- "Keywords" on page 479
- "Usage notes" on page 479

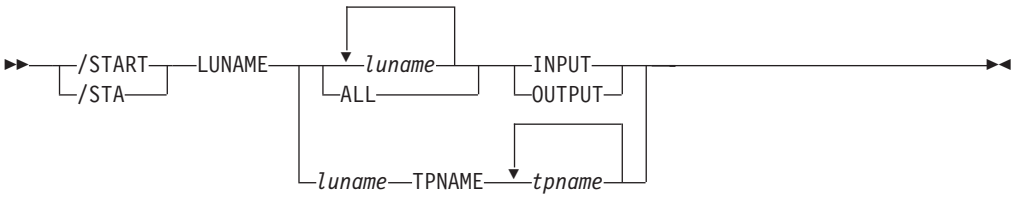
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 212. Valid environments for the START LUNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
INPUT	X		X
LUNAME	X		X
OUTPUT	X		X

Syntax



Keywords

The following keywords are valid for the /START LUNAME command:

INPUT
Specifying the keyword INPUT starts an luname for any input and synchronous outbound activities. Specifying the parameter ALL with the keyword INPUT causes all future LU 6.2 inbound and synchronous output activities to be started as well.

OUTPUT
Specifying the keyword OUTPUT starts an luname for asynchronous outbound activities. Specifying the parameter ALL with the keyword OUTPUT causes all future LU 6.2 outbound asynchronous activities to be started as well.

Usage notes

Specifying neither INPUT nor OUTPUT is the same as specifying both INPUT and OUTPUT. The LU name is started for any input and both synchronous and asynchronous outbound activities. Specifying the parameter ALL in this case also causes the start of all future LU 6.2 inbound activities, outbound synchronous, and asynchronous activities.

A network-qualified LU name is optional for the LUNAME keyword. If the LU name is not network-qualified and no TP name is specified, all the network-qualified LU names whose LU name matches the LU name specified are also started.

/START LUNAME TPNAME starts a particular tpname of an luname. The keyword OUTPUT is the default for this command.

If the specified resource does not exist, a structure is created to retain the status.

Related reference:

Command keywords and their synonyms (Commands)

/START MADSIOT command

Use the /START MADSIOT command to specify the MADS I/O timing function.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 213. Valid environments for the /START MADSIOT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
MADSIOT	X	X	

Syntax

Usage notes

The /START MADSIOT command is valid only after the long busy handling function is disabled for a link failure, a structure failure, or a rebuild failure. When the /START MADSIOT command completes normally, one of the two following messages is returned to the operator's console:

- `DFS1728E START MADSIOT COMMAND FAILED RSN=rrr`
- `DFS1727I MADSIOT TIMING FUNCTION RESUMED SUCCESSFULLY`

The purpose of this command is to enable MADS I/O Timing function. If MADS I/O Timing list structure is not defined in DFSVSMxx, the command will be rejected. If MADS I/O Timing function is already enabled, the command will be ignored. If MADS I/O Timing function is not enabled and all sharing partners successfully connect to MADS I/O Timing list structure on the coupling facility, the command will complete successfully; if any sharing partners fails to connect to MASD I/O Timing list structure, the command will fail.

The output of the /STA MADSIOT command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS0023I, DFS0007I, DFS1270I, DFS1727I, DFS1552A, DFS1728E

/START MSNAME command

Use the /START MSNAME command to specify the logical link path that is to be started. The MSNAME keyword can be generic.

Subsections:

- "Environment"
- "Syntax"
- "Examples"

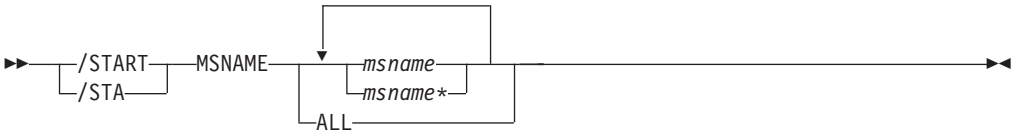
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 214. Valid environments for the /START MSNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
MSNAME	X		X

Syntax



Examples

The following is an example of the /START MSNAME command:

Entry ET:

```
/START MSNAME CHICAGO
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: A logical link path associated with the name CHICAGO is started.

Related reference:

"UPDATE MSNAME command" on page 717

/START NODE command

Use the /START NODE command to allow IMS to accept logons from, or initiate logons to, terminals attached to VTAM without actually initiating a session with the terminal. When a stopped node is started by using the /START NODE command, the terminal can log on to IMS or IMS can initiate a session with the terminal by using the /OPNDST NODE command.

The /START NODE command operates only if the node is disconnected, idle, and stopped. In addition to resetting the STOPPED status, the command resets MFSTEST mode, exclusive mode, DEACT status, and conversational mode by terminating the conversations.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 483

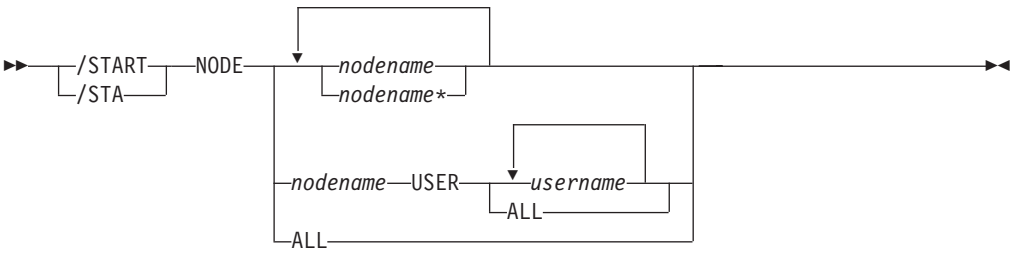
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 215. Valid environments for the /START NODE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
NODE	X		X
USER	X		X

Syntax



Usage notes

The /START NODE command is valid for existing nodes only (including any temporary nodes present in the system).

There is no need to issue a /DEQUEUE command for Fast Path messages. When the response mode is reset for a NODE or USER destination of Fast Path messages by either a /STOP NODE or a /STOP USER command, the Fast Path messages are discarded when the accompanying /START NODE or /START USER command is issued. In this case, IMS writes a X'67D0' Subtype 11 trace log record.

The /START NODE nodename USER username command applies to ISC sessions only, and it is used to start a half session allocated to USER *username* for NODE *nodename*. The USER keyword when used with the NODE keyword affects the specified half-session. When the USER keyword is omitted, all half-sessions of the specified node are affected.

Restrictions for using NODE and USER parameters together:

- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- /START NODE USER commands are valid for ISC, LUP, and 3600 nodes only.

The NODE parameter can be generic if the USER keyword is not present. The generic parameter specifies nodes that already exist.

The /START NODE command no longer resets test mode and preset mode, because these statuses are no longer significant and therefore no longer carried across logon or restart. MFSTEST mode (at the node level) and exclusive mode are still reset.

If global resource information is kept in Resource Manager, the /START NODE command allows a node to log on to any IMS in the IMSplex and resets MFSTEST mode and exclusive mode. If the node no longer has significant status, it is deleted from Resource Manager.

If a node in conversational mode receives the message, `DFS058I START COMMAND COMPLETED EXCEPT` when a /START NODE command is issued, it is possible that the conversation is `INUSE` by some other process. This is a temporary condition; you can reissue the /START NODE command.

Examples

The following is an example of the /START NODE command:

Entry ET:

```
/START NODE HARRY
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: The physical terminal that is associated with the node HARRY is started.

/START OLDS command

The /START OLDS command indicates that either a previously stopped OLDS is to be started or that IMS is to add a new OLDS log data set.

If a new OLDS is being added, olds# is an OLDS identifier that is defined by the DFSMDA macro specification. If in dual mode, both primary and secondary OLDSs are started. olds# must be 00-99.

When using /START OLDS, an OLDS must be defined in the DFSMDA macro, even if it is allocated in JCL.

Subsections:

- "Environment" on page 484
- "Syntax" on page 484
- "Examples" on page 484

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 216. Valid environments for the /START OLDS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
OLDS	X	X	X

Syntax

Examples

The following is an example of the /START OLDS command:

Entry ET:

```
/START OLDS 09
```

Response ET:

```
DFS058I  START COMMAND IN PROGRESS
```

Explanation: OLDS data set DFSOLP09 (DFSOLS09) will be started for logging.

/START OTMA command

Use the /START OTMA command to cause IMS to join the z/OS cross-system coupling facility (XCF) group for the IMS Open Transaction Manager Access (OTMA).

Subsections:

- "Environment"
- "Syntax" on page 485
- "Usage notes" on page 485
- "Examples" on page 485

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 217. Valid environments for the /START OTMA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
OTMA	X		X

Syntax

Usage notes

/START OTMA command processing is as follows:

1. IMS joins the XCF group.
2. Following a successful Client-Bid, IMS sends an ACK message to the OTMA client.
3. IMS begins sending all Commit-then-Send (commit mode 0) output messages to the OTMA client.

The NOCHECK option specifies that the command is not recovered during emergency restart for OTMA.

Examples

The following is an example of the /START OTMA command:

Entry ET:

```
/STA OTMA
```

Response ET:

```
DFS2360I 14:02:53 XCF GROUP JOINED SUCCESSFULLY. SYS3
DFS058I 14:02:53 START COMMAND COMPLETED   SYS3
DFS996I *IMS READY*  SYS3
```

/START PGM command

Use the /START PGM command to specify the application program that is to be started.

This command also clears the indicator preventing a program from scheduling when I/O prevention has not completed. The integrity of a GSAM database residing on DASD can be affected if I/O prevention has not been done on a failing active system.

/START PGM does not start a CPI Communications driven transaction program.

Subsections:

- "Environment"
- "Syntax" on page 486
- "Equivalent IMS type-2 commands" on page 486
- "Examples" on page 486

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 218. Valid environments for the /START PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
PGM	X	X	X

Syntax

Equivalent IMS type-2 commands

The following table shows variations of the /START PGM command and the IMS type-2 commands that perform similar functions.

Table 219. Type-2 equivalents for the /START PGM command

Task	/START PGM command	Similar IMS type-2 command
Starts program scheduling.	/START PGM *pgmname*	UPDATE PGM NAME(*pgmname*) START(SCHD)

Examples

The following are examples of the /START PGM command:

Example 1 for /START PGM command

Entry ET:

```
/START PROGRAM ALL
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: All application programs are started.

Example 2 for /START PGM command

Entry ET:

```
/START PROGRAM APPLETRE
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: Application program APPLETRE is started.

Related reference:

"UPDATE PGM command" on page 755

/START REGION command

Use the /START REGION command to specify the set of message processing region JCL to be passed to z/OS. If no member name is specified, the default member name is used

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 488

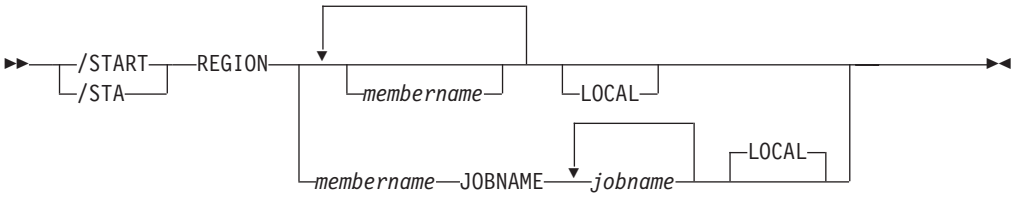
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 220. Valid environments for the /START REGION command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
LOCAL	X	X	
REGION	X	X	X

Syntax



Usage notes

IMS dependent regions of the same type (MPP, BMP, or IFP) can share a PROCLIB member containing the startup JCL for the type of region. Use the JOBNAME or LOCAL keywords of the /START REGION command to enable IMS to set (or override) the IMS ID for the dependent region to match the IMS ID of the IMS that processes the command.

Restriction: The JCL for the region to be started must include the IMSID= execution parameter.

Use the JOBNAME keyword to override the job name on the JOB statement of the default or specified JCL member for a dependent region.

If you specify the LOCAL keyword, IMS overrides the symbolic IMSID parameter in the JCL of the default or specified member. LOCAL is the default if you specify the JOBNAME keyword.

When the LOCAL or JOBNAME keywords are specified on the /START REGION command, the PROCLIB member must be a job that runs a procedure to start the dependent region. The procedure cannot be an instream procedure. For example, suppose that the /START REGION command is entered in one of the following formats:

```
/START REGION member_name LOCAL
/START REGION member_name JOBNAME job_name
/START REGION member_name JOBNAME job_name LOCAL
```

In these instances, *member_name* is a job that runs a procedure to start the dependent region and has the following format:

```
//job_name JOB ... (parameters)...
//      EXEC proc_name,
//       IMSID=xxxx
```

The operator can start more dependent regions than were specified in the IMS system definition or the EXEC parameter, up to 999. A request to start more regions than the system-definition value (but less than or equal to 999) might be rejected if resources are not available.

The /START REGION command is not mirrored on the XRF alternate subsystem. You must enter this command on the alternate subsystem if you want it to affect the alternate subsystem.

Examples

The following are examples of the /START REGION command:

Example 1 for /START REGION command

Entry ET:

```
/START REGION
```

Response ET:

```
DFS058I  START COMMAND IN PROGRESS
```

Response ET:

```
DFS551I IFP|MESSAGE|BATCH  REGION XXXXXXXX STARTED. ID=yy TIME=zzzz
CLASSES=xxx,xxx,xxx,xxx
```

Explanation: One message region or batch region (ID=yy) is started at TIME=zzzz. The transactions associated with the classes listed in the response can now be scheduled.

Example 2 for /START REGION command

Entry ET:

```
/START REGION IMSWT000
```

Response ET:

```
DFS058I  START COMMAND IN PROGRESS
```

Explanation: The JCL stored as member IMSWT000 is used to start the spool SYSOUT utility for the data sets associated with the spool line corresponding to the IMSWT000 procedure.

Example 3 for /START REGION command

Entry ET:

```
/START REGION MEMABC
```

Response ET:

```
DFS058I  START COMMAND IN PROGRESS
```

Response ET:

```
DFS551I IFP|MESSAGE|BATCH  REGION XXXXXX STARTED. ID=yy TIME=zzzz
CLASSES=xxx,xxx,xxx,xxx
```

Explanation: The JCL stored as member XXXXXX is used to start a message processing region or batch message processing region with the classes specified by the EXEC statement parameters in MEMABC.

Related reference:

PRDR= through QTU= procedure parameters (System Definition)

/START RTC command

Use the /START RTC command to specify the Fast Path routing codes to be activated and to allow transactions associated with the routing codes to be processed.

Subsections:

- "Environment"
- "Syntax"
- "Equivalent IMS type-2 commands"
- "Examples" on page 490

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 221. Valid environments for the /START RTC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
RTC	X		X

Syntax

Equivalent IMS type-2 commands

The following table shows variations of the /START RTC command and the IMS type-2 commands that perform similar functions.

Table 222. Type-2 equivalents for the /START RTC command

Task	/START RTC command	Similar IMS type-2 command
Starts queuing to a Fast Path routing code.	/START RTC *rtcname*	UPDATE RTC NAME(*rtcname*) START(Q)

Examples

The following is an example of the /START RTC command:

Entry ET:

```
/START RTCODE ALL
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: All the Fast Path routing codes are activated. Transactions associated with these routing codes can now be processed.

Related reference:

"UPDATE RTC command" on page 782

/START SB command

Use the /START SB command to dynamically allow sequential buffering. This command does not affect sequential buffering applications scheduled before this command was issued.

Subsections:

- "Environment"
- "Syntax"
- "Examples" on page 491

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 223. Valid environments for the /START SB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SB	X	X	

Syntax

Examples

The following is an example of the /START SB command:

Entry ET:

```
/START SB
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Entry ET:

```
/DISPLAY POOL DBAS
```

Response ET:

```
SEQUENTIAL BUFFERING:  STATUS = NOT-STOPPED
MAX      N.A.  FREE   N.A.   CURR   160K  HIGH  320K
DATABASE BUFFER POOL: SIZE   67584
REQ1       0 REQ2     0 READ     0 BISAM     0 WRITES     0
KEYC       0 LCYL     0 PURG     0 OWNRR     0 ERRORS 00/00
DATABASE BUFFER POOL:  BSIZE  12288
RRBA     0 RKEY     0  BFALT    0 NREC  0 SYN PTS     0
NMBUFS  29 VRDS     0   FOUND   0 VWTS  0  ERRORS 00/00
DATABASE BUFFER POOL:  BSIZE  356352
RRBA     0 RKEY     0  BFALT    0 NREC  0 SYN PTS     0
NMBUFS  29 VRDS     0  FOUND    0 VWTS  0  ERRORS 00/00
*86253/104547*
```

Explanation: Sequential buffering is started.

/START SERVGRP command

Use the /START SERVGRP command to start communications between the entering service group and the service group at the other site in an Remote Site Recovery (RSR) complex.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes" on page 492

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 224. Valid environments for the /START SERVGRP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SERVGRP	X	X	X

Syntax

Usage notes

If the subsystem is not currently identified to the Transport Manager Subsystem (TMS), an attempt to identify precedes an attempt to connect to the other subsystem. /START SERVGRP is supported on an active IMS subsystem and an RSR tracking subsystem.

The /START SERVGRP command is not normally needed for an active subsystem because the logger normally attempts to identify to TMS at each OLDS switch to establish connections with the other subsystem. However, the operator might want to trigger this process between OLDS switches; for instance, if a network outage between the active and tracking sites has been repaired and the operator does not want to wait until the next OLDS switch to re-establish communications.

Successful completion of the syntax checking of the /START SERVGRP command results in the `DFS058 START COMMAND COMPLETED` message, although processing of the command continues asynchronously.

Examples

The following is an example of the /START SERVGRP command:

Entry ET (at the active site):

```
/START SERVGRP
```

Response ET (to the active subsystem):

```
DFS058 START COMMAND COMPLETED
```

Explanation: Communications between the subsystem at the active site and the subsystem at the RSR tracking site are started.

/START SLDSREAD command

The /START SLDSREAD command indicates whether IMS is enabled to retrieve records from both a system log data set (SLDS) and OLDS or OLDS only. The default is that SLDSREAD is enabled.

Subsections:

- "Environment"
- "Syntax" on page 493

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 225. Valid environments for the /START SLDSREAD command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SLDSREAD	X	X	X

Syntax

/START SUBSYS command

Use the /START SUBSYS command to specify the external subsystem to which IMS is to connect.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 494

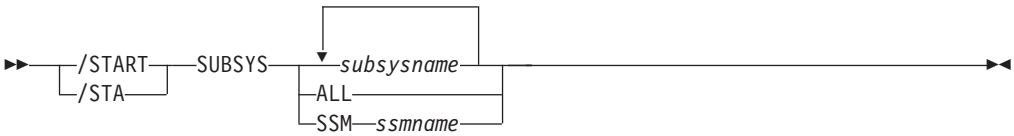
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 226. Valid environments for the /START SUBSYS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SSM	X	X	X
SUBSYS	X	X	X

Syntax



Keywords

The following keywords are valid for the /START SUBSYS command:

SSM

Allows external subsystem connection processing to occur even though the option was not requested when IMS was started.

The syntax and usage of the SSM keyword are the same as the SSM= EXEC parameter that can be specified on the IMS startup JCL. The SSM keyword is followed by a 1 to 4 character identifier. IMS concatenates the SSM identifier to the IMSID to create an SSM PROCLIB member name. The SSM PROCLIB member is then used for external subsystem processing.

The SSM keyword is not valid if either of the following conditions apply:

- The SSM= keyword is specified in the EXEC parameters of the startup JCL.
- The /START SUBSYS SSM command has been previously issued.

Usage notes

This command can also be used to dynamically reconfigure existing subsystem definitions. The installation can start IMS with the subsystem PROCLIB member defining one subsystem. The PROCLIB member can then be changed or added to. The operator can then /STOP the existing subsystem connections or only the one that has changed. By issuing the /START SUBSYS command, IMS will pick up the new or changed definitions and attempt to connect to those subsystems.

The /START SUBSYS ALL command connects IMS to all external subsystems. Also, the SSM keyword can be used with the /START SUBSYS command.

If the subsystem connection was abnormally terminated, IMS puts the connection in a stopped state. In this instance, the /START command must be used to reestablish the connection.

Examples

The following is an example of the /START SUBSYS command:

Entry ET:

```
/START SUBSYS ABC
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: IMS has established a connection to the requested subsystem. It is likely that an external subsystem (not CCTL) connection message will be received at this time. If this is not the case, the /DISPLAY command can be used.

/START SURV command

Use the /START SURV command in an XRF environment to start the operation of the IMS surveillance function.

Subsections:

- "Environment"
- "Syntax" on page 495
- "Usage notes" on page 495

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 227. Valid environments for the /START SURV command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SURV	X		X

Syntax

Usage notes

When surveillance is on for a function, potential failures of the active system are detected. Based on information from surveillance, the alternate system either requests a takeover or informs the operator of the potential failure. The following are the surveillance functions to be started:

ALL Same as specifying LNK, RDS, and LOG.

LNK ISC link.

LOG System log.

RDS Restart data set.

The surveillance function is generally started during IMS system definition by using the /START SURVEILLANCE control statement. ALL is the default.

/START THREAD command

Use the /START THREAD command to specify the set of message processing region JCL to be passed to z/OS. If no member name is specified, the default member name is used.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes" on page 496

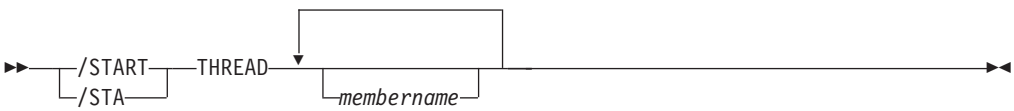
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 228. Valid environments for the /START THREAD command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
THREAD	X	X	

Syntax



Usage notes

The /START THREAD command is used only for DEDB utility regions. BMP regions are started by JCL. CCTL threads are started automatically at connection: first to the MINTHREAD value, and later (on demand) to MAXTHREAD value.

/START TMEM command

Use the /START TMEM command to cause IMS to send an Open Transaction Manager Access (OTMA) command to OTMA clients to request that input resume for the specified transaction membername. IMS then resumes sending output to the OTMA client.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Examples" on page 498

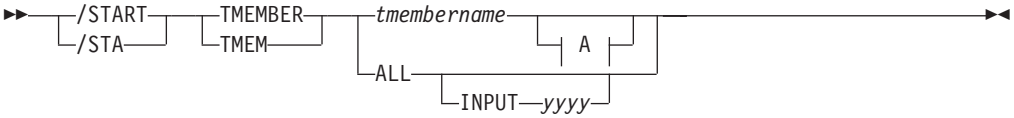
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 229. Valid environments for the /START TMEM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
TMEM	X		X
TPIPE	X		X

Syntax



A:

Keywords

The following keywords are valid for the /START TMEM command:

INPUT
Specifies the maximum number of concurrent input messages from the

OTMA member. OTMA monitors the growth of the input messages from the member. A DFS1988W warning message is sent to the system console to indicate that the input messages reached 80% of the limit and the message is issued every 5% thereafter. When the maximum is reached, a DFS1989E error message is sent to the console. Any subsequent OTMA input message from this member will be rejected with the OTMA sense code X'30'.

When the number of input messages (transaction instance blocks (YTIBs)) reaches the maximum, the /DISPLAY command shows FLOOD status under the user-status column. FLOOD status is relieved when the input messages are processed and recede to 50% or less of the maximum value or a /START TMEMBER INPUT command is issued with a higher maximum value specified.

The input value can be specified 9 - 9999. If the value is 0, OTMA deactivates the message flood detection. If the value is 1 - 200, it is treated as 200. If it is over 9999, it is rejected.

If this keyword is specified with the ALL keyword, OTMA uses this new limit instead of the system default of 8000 to monitor the total active input message count (YTIB) for send-then-commit transactions from all the OTMA members. If the total number reaches this specified limit, OTMA will issue a DFS4388W message to IMS MTO and system console, and an OTMA protocol message with the warning status is sent to all the OTMA clients. If the condition is relieved, OTMA issues a DFS0798I message to IMS MTO and system console, and an OTMA protocol message with the good status is sent to all the OTMA clients.

TPIPE

When it is used with the /START command, TPIPE causes IMS to send an OTMA command to its OTMA clients to request that the input resume for the specified transaction pipe. IMS then resumes sending output to the OTMA client. If the member specified is a super member, output is resumed for the super member's transaction pipe, but no OTMA command is sent. If the member specified is a regular member whose hold queue output is managed by a super member, IMS resumes output for the specified member's transaction pipe and it also resumes output for the super member's transaction pipe. An OTMA command is sent to the regular member's OTMA client. Output is only resumed on the IMS that processes the command. If output cannot be resumed for both the regular member's transaction pipe and the super member's transaction pipe, it is not resumed for either transaction pipe. The DFS058I COMMAND COMPLETED EXCEPT message is issued with the name of the regular member for which output could not be resumed.

Restriction: If a transaction pipe has a resynchronization pending status, IMS does not start the transaction pipe.

TIMEOUT

Sets the timeout interval for acknowledgments to commit-then-send (CM0) and send-then-commit (CM1) output messages.

After OTMA delivers a send-then-commit output message that used synclevel=confirm or synclevel=syncpt, OTMA expects an acknowledge (ACK or NAK) message from the OTMA client, such as IMS Connect. If the acknowledge message is not received by IMS within the specified

timeout value, OTMA will take the timeout action to abort the transaction. This prevents IMS dependent region from getting the WAITSYNPT or WAIT/RRS status for long time.

The timeout interval must be 0 - 255 seconds, with a default value of 120 seconds. When 0 is specified, the timeout function is disabled.

Examples

The following are examples of the /START TMEM command:

Example 1 for /START TMEM command

Entry ET:

```
/STA TMEMBER CLIENT1 TPIPE TPIPESY
```

Response ET:

```
DFS058I 15:39:40 START COMMAND COMPLETED   SYS3
DFS996I *IMS READY*  SYS3
```

Example 2 for /START TMEM command

Entry ET:

```
 /START TMEMBER HWS1 INPUT 2000
```

Response ET:

```
DFS058I START COMMAND COMPLETED
```

Explanation: The maximum concurrent input message count for the OTMA member HWS1 has been set to 2000. Based on the number specified, IMS OTMA will monitor the growth of the input messages to prevent a message flood condition.

/START TRAN command

Use the /START TRAN command to specify the transactions to be started.

Subsections:

- "Environment"
- "Syntax" on page 499
- "Usage notes" on page 499
- "Equivalent IMS type-2 commands" on page 500
- "Examples" on page 500

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

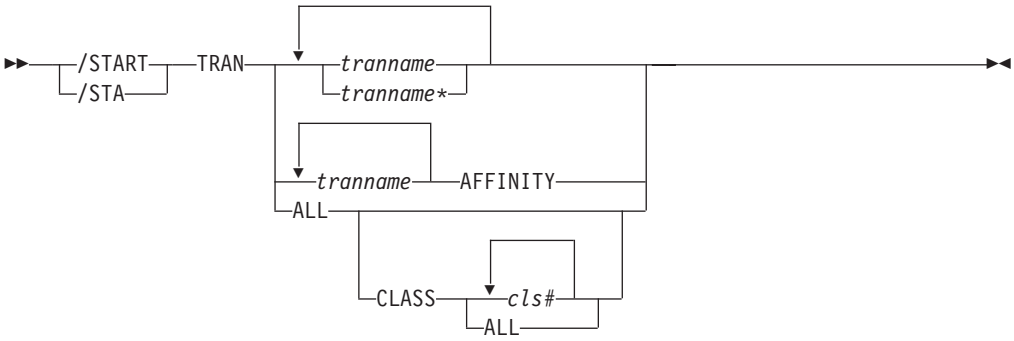
Table 230. Valid environments for the /START TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
AFFINITY	X		X

Table 230. Valid environments for the /START TRAN command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
CLASS	X		X
TRAN	X		X

Syntax



Usage notes

The /START TRAN ALL CLASS *cls#* command causes all transactions associated with the specified class to be started.

If a transaction that has messages on the suspend queue is started, the suspend queue associated with the transaction is automatically transferred to the normal queue.

The TRAN keyword can be generic where the generic parameter specifies transactions that already exist.

You can use the /START TRAN *tranname* AFFINITY command to start a local affinity transaction queue in a shared-queues environment. Starting a local affinity transaction queue in a shared-queues environment causes this IMS system to be notified when transaction messages that have an affinity name of this IMS system's IMSID or RSENAME (XRF) appended to the transaction name are on the shared queues for processing. This affinity registration is in addition to the normal transaction registration (registration with the transaction name appended with blanks).

When a transaction is started with affinity, the affinity registration status cannot be removed and the QUERY TRAN command always shows the affinity status. The following /STOP TRAN *tranname* command therefore stops both normal and affinity registration. The subsequent /START TRAN *tranname* command (with or without the AFFINITY keyword) always performs both normal and affinity registration.

The affinity status is lost across IMS cold starts. If the DFSMSCE0 user message routing exit is used to set local affinity for an input transaction message in a shared-queues environment and the IMS control region is stopped and subsequently cold started, you must issue the /START TRAN AFFINITY command

or the UPDATE TRAN NAME(*tranname*) START(SCHD) OPTION(AFFIN) command. These commands register the transaction with affinity status so that the messages will be processed.

At the end of cold start processing, when shared-queues informs are done, the inform for the transaction is done without affinity to the local IMSID. Because local affinity is set by the exit and is not part of the transaction definition, it is not maintained across a cold start. As a result, the message with local affinity cannot be scheduled. The /START TRAN *tranname* AFFINITY command issues an inform for the transaction with affinity to the local IMSID.

The /START TRAN *tranname* AFFINITY command does not support generic transaction names of ALL in the parameter.

In a shared-queues environment, the /START TRAN command will result in IMS registering interest for the transaction, which indicates that the transaction can be scheduled at the IMS. A /START TRAN ALL command does not register transactions that are already registered to CQS.

Equivalent IMS type-2 commands

The following table shows variations of the /START TRAN command and the IMS type-2 commands that perform similar functions.

Table 231. Type-2 equivalents for the /START TRAN command

Task	/START TRAN command	Similar IMS type-2 command
Starts a transaction.	/START TRAN	UPDATE TRAN NAME(*tranname*) START(Q,SCHD,SUSPEND)

Examples

The following are examples of the /START TRAN command:

Example 1 for /START TRAN command

Entry ET:

```
/START TRAN ALL CLASS 6
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: All transactions associated with class 6 are started.

Example 2 for /START TRAN command

Entry ET:

```
/START TRAN PIT, SEED
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Explanation: Transactions PIT and SEED are started.

Example 3 for /START TRAN command

Entry ET:

```
/START TRAN APOL12 AFFINITY
```

Response ET:

```
DFS058I START COMMAND COMPLETED
```

Explanation: Transaction APOL12 is started with affinity.

Related reference:

"UPDATE TRAN command" on page 795

/START TRKARCH command

Use the /START TRKARCH command to indicate that the RSR tracking subsystem is to initiate a request to start the automatic archiving of the tracking log data sets.

This keyword allows the user to start automatic archive after it has been terminated following archive data set full conditions.

Successful completion of the syntax checking of the /START TRKARCH command results in the `DFS058I START COMMAND COMPLETED` message, although processing of the command continues asynchronously.

Subsections:

- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 232. Valid environments for the /START TRKARCH command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
TRKARCH	X	X	X

Syntax

/START USER command

Without the NODE keyword, USER specifies the ISC user or the dynamic user to start. The USER parameter can be generic where the generic parameter specifies users that already exist.

Subsections:

- "Environment" on page 502
- "Syntax" on page 502
- "Usage notes" on page 502

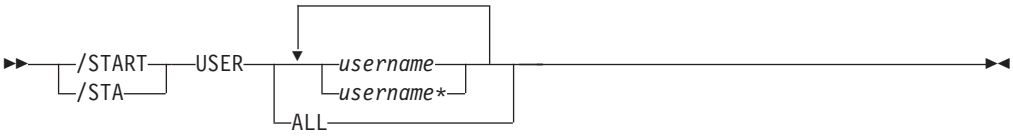
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 233. Valid environments for the /START USER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
USER	X		X

Syntax



Usage notes

The /START USER command applies only to users that are stopped and signed off. The /START USER command also terminates any active conversations before starting the user.

The /START USER command removes the user from MFSTEST mode and exclusive mode. If the USER structure is temporary and the status conditions that caused the creation of the structure have been reset, the temporary user is deleted at the next simple checkpoint.

For ISC users, the user is started and made available for allocation.

For dynamic users, the user is started and made available for signon. If the dynamic user was in either Fast Path input or output response mode, and the Fast Path input or output response mode is reset by the /STOP USER and /START USER commands issued in sequence, IMS writes a X'67D0' Subtype 11 trace log record.

The /START USER command no longer removes the user from test mode, and preset mode. MFSTEST mode can now be associated with the node and with the user. MFSTEST mode (at the user level) and exclusive mode are still reset. The other statuses are no longer significant and therefore not carried across signon or restart.

If global resource information is not kept in Resource Manager, the /START USER command allows a user to signon to the local IMS. If global resource information is kept in Resource Manager, the /START USER command allows a user to signon to any IMS in the IMSplex.

If a user in conversational mode receives the message, `DFS058I START COMMAND COMPLETED EXCEPT` when a /START USER command is issued, it is possible that the conversation is `INUSE` by some other process. This is a temporary condition; you can reissue the /START USER command.

Examples

The following is an example of the /START USER command:

Entry ET:

```
/DISPLAY USER IMSUS01 IMSUS02
```

Response ET:

```
  USER     ENQCT DEQCT   QCT
  IMSUS01      0     0     0 STOPPED
  IMSUS02      0     0     0 STOPPED
  *91091/111727*
```

Entry ET:

```
R 38,/START USER IMSUS01
```

Response ET:

```
DFS058I 11:19:05 START COMMAND COMPLETED
```

Entry ET:

```
/DISPLAY USER IMSUS01 IMSUS02
```

Response ET:

```
USER     ENQCT DEQCT   QCT
IMSUS01      0     0     0
IMSUS02      0     0     0 STOPPED
*91091/113038*
```

Entry ET:

```
/START USER APPLE*
```

Response ET:

```
DFS3633 11:19:35 GENERIC PARAMETER RESOURCES NOT FOUND, NO ACTION TAKEN
```

/START VGR command

The /START VGR command causes the IMS subsystem to join a VTAM generic resource group. The command is rejected if the VTAM ACB is closed (usually the result of a /STOP DC command).

Subsections:

- "Environment"
- "Syntax" on page 504
- "Keywords" on page 504

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

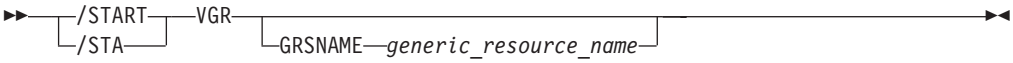
Table 234. Valid environments for the /START VGR command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
GRSNAME	X	X	X

Table 234. Valid environments for the /START VGR command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
VGR	X		X

Syntax



Keywords

The following keywords are valid for the /START VGR command:

The GRSNAME keyword allows you to specify the generic resource name if the IMS subsystem does not have one. The IMS subsystem already has a generic resource name if it has the GRSNAME= keyword specified on its EXEC statement.

The GRSNAME is ignored if GRSNAME= was specified on the EXEC statement, or if it was already specified on a previous /START VGRS command. If GRSNAME= was not specified on the EXEC statement or any previous /START VGRS command, then all VTAM sessions must be terminated prior to executing the /START command with GRSNAME specified.

/START WADS command

The /START WADS command indicates that either a previously stopped WADS is to be started or that IMS is to add a new WADS to the pool of available WADSs.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 505
- "Usage notes" on page 505

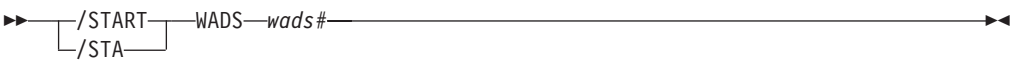
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 235. Valid environments for the /START WADS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
WADS	X	X	X

Syntax



Keywords

The following keywords are valid for the /START WADS command:

wads#
If a new WADS is being added, *wads#* is a WADS identifier that is defined by the DFSMDA macro specification. *wads#* must be 0-9.

Usage notes

When using the /START WADS command, a WADS must be defined in the DFSMDA macro, even if it is allocated in JCL.

/START XRCTRACK command

The /START XRCTRACK command results in calls to the log router to initiate or terminate XRC tracking. It is only valid on a tracking IMS system.

Subsections:

- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 236. Valid environments for the /START XRCTRACK command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
XRCTRACK	X	X	X

Syntax

Chapter 23. /STOP commands

The /STOP commands stop the sending, receiving, or queuing of output messages to a particular communication line, terminal, user, or logical path.

You can also use these commands to stop the scheduling or queuing of messages containing a specific transaction code, the execution of a specific program, or the use of a given database.

For VTAM nodes, the currently connected terminal is disconnected. All further logons are rejected until the node is the subject of a /START or /RSTART command.

The /STOP command validity checks all parameters entered by the terminal operator. If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and the /STOP command processes the rest of the parameters.

The /STOP command can be used to reset conditions previously established by the /START, /RSTART, /PSTOP, /PURGE, or /MONITOR commands.

These commands can be issued to an IMSplex using the Batch SPOC utility.

- "/STOP ADS command"
- "/STOP APPC command" on page 508
- "/STOP AREA command" on page 509
- "/STOP AUTOARCH command" on page 513
- "/STOP BACKUP command" on page 513
- "/STOP CLASS command" on page 514
- "/STOP DATAGRP command" on page 515
- "/STOP DB command" on page 516
- "/STOP DC command" on page 520
- "/STOP DESC command" on page 521
- "/STOP LINE command" on page 522
- "/STOP LTERM command" on page 523
- "/STOP LUNAME command" on page 525
- "/STOP MADSIOT command" on page 526
- "/STOP MSNAME command" on page 527
- "/STOP NODE command" on page 528
- "/STOP OLDS command" on page 530
- "/STOP OTMA command" on page 531
- "/STOP PGM command" on page 532
- "/STOP REGION command" on page 533
- "/STOP RTC command" on page 541
- "/STOP SB command" on page 541
- "/STOP SERVGRP command" on page 542
- "/STOP SLDSREAD command" on page 543
- "/STOP SUBSYS command" on page 544
- "/STOP SURV command" on page 545
- "/STOP THREAD command" on page 546
- "/STOP TMEM command" on page 549
- "/STOP TRAN command" on page 551
- "/STOP USER command" on page 553
- "/STOP VGR command" on page 555
- "/STOP WADS command" on page 555
- "/STOP XRCTRACK command" on page 556

/STOP ADS command

The /STOP ADS command specifies the area data set to be closed and deallocated.

© Copyright IBM Corp. 1974, 2015

The AREA is not stopped as long as at least one data set in the AREA remains open. /STOP ADS is rejected if the specified ADS is the last data set available in the AREA.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 237. Valid environments for the /STOP ADS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
ADS	X	X	

Syntax

Usage notes

Although the /STOP ADS command has no option of LOCAL/GLOBAL, if the DEDB area is shared at the block level, the response is the same as if GLOBAL were specified.

The output of the /STOP ADS command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS140I, DFS0488I, DFS0666I, DFS1407I, DFS3720I, DFS3721I, DFS3771I

/STOP APPC command

The /STOP APPC command instructs IMS to stop scheduling transactions from LU 6.2 devices. The /STOP APPC command can be used in a transient stopped state. It causes remote LU 6.2 devices to receive a sense code of TP_Not_Available_No_Retry. This is likely to lead to further attempts to access IMS.

Subsections:

- "Environment" on page 509
- "Syntax" on page 509
- "Keywords" on page 509
- "Usage notes" on page 509

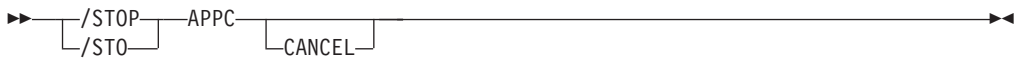
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 238. Valid environments for the /STOP APPC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
APPC	X		X
CANCEL	X	X	X

Syntax



Keywords

The following keyword is valid for the /STOP APPC command:

CANCEL
Causes APPC/MVS to initiate a shutdown request when a long stopped period is anticipated, for example, at the end of the day's processing. All remote LU 6.2 devices receive a sense code of TP_Not_Available_No_Retry. The remote LU 6.2 devices stop trying to access this application.

The /STOP APPC CANCEL command sets the desired status to CANCEL according to responses from APPC/MVS. If the desired status is DISABLED, then IMS rejects /STOP APPC CANCEL when it is entered.

Note: The sense code returned to the LU 6.2 remote device for an incoming ATTACH to a stopped APPC/IMS system is determined by APPC/MVS, and it might differ from release to release. In general, the remote LU 6.2 application should wait for a period of time after rejection before any attempts to reestablish a session with IMS.

Usage notes

The /STOP APPC command sets the desired status to STOPPED. The current status is set to STOPPED or FAILED according to the response from APPC/MVS.

/STOP AREA command

The /STOP AREA command specifies that the data sets associated with this area are closed.

Subsections:

- "Environment" on page 510
- "Syntax" on page 510
- "Keywords" on page 510
- "Usage notes" on page 512
- "Equivalent IMS type-2 commands" on page 512
- "Examples" on page 512

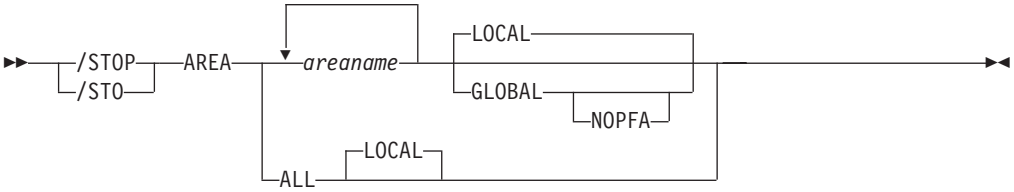
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 239. Valid environments for the /STOP AREA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
AREA	X	X	
GLOBAL	X	X	
LOCAL	X	X	
NOPFA	X	X	

Syntax



Keywords

The following keywords are valid for the /STOP AREA command:

GLOBAL
Specifies when an IRLM is active and that the command applies to all subsystems sharing the database or area.

The GLOBAL keyword and the ALL parameter are mutually exclusive. If both keywords are specified, the command is rejected. The GLOBAL keyword requires that IRLM be active. If IRLM is not active, the command is rejected. DBRC is informed that the database or area has been stopped and will update the RECON data set to indicate the stopped condition.

The GLOBAL command is processed by the IMS system where the command was initiated. This system will process the command locally and then request IRLM NOTIFY to route and process the command on sharing IMS systems.

If global DB or AREA status is maintained, the global status maintained in RM is also updated. The global status is set to STOPPED.

If the command is entered from OM API, the global status is updated by the command master IMS. If the command is not entered from OM API, the IMS that initiated the GLOBAL command updates the global status in RM

You must specify if the global area status must be maintained in RM. You can specify this during IMS initialization in either the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSAREA(Y)). You can also change it dynamically using the UPDATE IMS SET(PLEXPARM(GSTSAREA(Y))) command. If you do not specify that the global area status is to be maintained, then the GLOBAL keyword is processed as in prior releases, and the global status is not updated.

If global status in RM is successfully updated, message DFS0988I for RSRCTYPE=AREA is issued. If global status is not successfully updated, message DFS3308I is issued, indicating RM failure, and no command response lines are generated. Any RM error is traced to the OCMD trace table. Users can issue a QRY DB STATUS(GLOBAL) command to obtain the global status of the resources in RM.

The X'594C' log record for DEDB areas includes both global status and global command time stamp.

The GLOBAL keyword is not supported on an RSR tracking subsystem.

If the GLOBAL keyword on a command is entered from an OM API, the command should only be routed to one IMS system in the IMSplex. The IMS that receives the command from OM will make DBRC calls to update the RECON with GLOBAL status. It will also request IRLM NOTIFY to route and process the command on sharing IMS systems, and then process the command locally.

Messages produced on the NOTIFIED systems will appear only on the system console and will not be routed back to the OM API which originally entered the command.

If multiple IMS systems have been explicitly specified in the route list, the master IMS system will process the command as described previously. However, the non-master IMS systems, to which OM routes the command, will reject the command with the following return and reason code listed in the following table:

Table 240. Return and reason code for GLOBAL keyword issued from the OM API

Return code	Reason code	Meaning
X'00000004'	X'00001000'	The command contained the GLOBAL keyword and was routed to more than one IMS system in the IMSPLEX. The non-master IMS systems will reject this command when OM routes the command to them. The master IMS system will process this command and use IRLM NOTIFY to route and process the command on the non-master IMS systems. See the discussion under the GLOBAL keyword.

LOCAL
Specifies that the command only applies to the subsystem in which the command is entered. This command does not affect any other subsystem sharing the database or area. LOCAL is the default.

NOPFA
Specifies that DBRC is not notified that the database or area has changed status. You can use this keyword when you need to authorize the database for use after it is offline, for example, for offline utilities. By using this keyword, DBRC does not prevent further authorizations for the database or area. NOPFA can be specified only with the GLOBAL keyword.

Recommendation: Before restarting the database or area, issue this command without the NOPFA keyword to inform DBRC of the change in status for the database or area.

Usage notes

Use the /DISPLAY AREA command to determine if the area is stopped or closed. If the area is stopped, the area must be made available using the /START AREA command. In z/OS, all the data sets are deallocated. If the system processes a /STOP AREA command during HSSP processing, the area will be released after the current commit processing completes. Any image copy option in effect at /STOP time can affect the continued system operation. All virtual storage option (VSO) DEDB areas that are being stopped and that are in a z/OS data space are removed from the data space and updates are written out to DASD.

The output of the /STOP AREA command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. For commands that specify GLOBAL, only the command master returns the asynchronous messages to OM. When a command is processed with the LOCAL keyword, all IMS systems are able to return the asynchronous messages to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS140I, DFS170I, DFS0488I, DFS0666I, DFS1407I, DFS3062I, DFS3342I, DFS3720I, DFS3824I

/STOP AREA is not supported on an RSR tracking subsystem.

While the database is being quiesced, this command cannot be processed successfully.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP AREA command and the IMS type-2 commands that perform similar functions.

Table 241. Type-2 equivalents for the /STOP AREA command

Task	/STOP AREA command	Similar IMS type-2 command
Stops an area.	/STOP AREA *areaname*	UPDATE AREA NAME(*areaname*) STOP(SCHD)

Examples

The following is an example of the /STOP AREA command:

Entry ET:

```
/STOP AREA DB1AREA0 DB1AREA1
```

Response ET:

```
DFS058I  STOP COMMAND IN PROGRESS
DFS0488I  STOP COMMAND COMPLETED.  AREA=DB1AREA0
DFS0488I  STOP COMMAND COMPLETED.  AREA=DB1AREA1
```

Explanation: The DEDB areas DB1AREA0 and DB1AREA1 are stopped for processing.

Related concepts:

Maintaining global information for databases, DEDB areas, and transactions (System Administration)

Related reference:
"UPDATE AREA command" on page 629

/STOP AUTOARCH command

The /STOP AUTOARCH command specifies that automatic archiving is to be stopped.

Subsections:

- "Environment"
- "Syntax"
- "Examples"

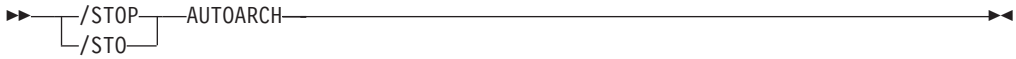
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 242. Valid environments for the /STOP AUTOARCH command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
AUTOARCH	X	X	X

Syntax



Examples

The following is an example of the /STOP AUTOARCH command:

Entry ET:

```
/STOP AUTOARCH
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: Automatic archiving is stopped.

/STOP BACKUP command

The /STOP BACKUP command terminates the alternate system in an XRF environment. This command must be entered on the alternate system. The ABDUMP keyword results in a dump of the alternate system.

Subsections:

- "Environment" on page 514
- "Syntax" on page 514

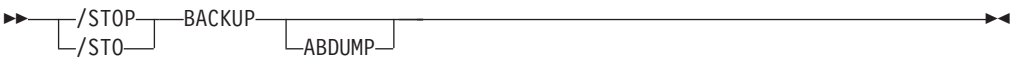
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 243. Valid environments for the /STOP BACKUP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
ABDUMP	X	X	X
BACKUP	X		X

Syntax



/STOP CLASS command

The /STOP CLASS command prevents further scheduling of application programs for the designated class.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 515

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 244. Valid environments for the /STOP CLASS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
CLASS	X		X

Syntax

Usage notes

All regions currently handling transactions assigned to the specific class are allowed to run until the limit count is reached (MPPs) or the input queue contains no more messages (BMPs and MPPs).

The region is not allowed to wait for the next message (wait-for-input mode). Instead a QC status code (no more messages) is returned to the application (MPPs).

If the region is already scheduled and waiting for the next message (wait-for-input mode) when the command is entered, the region is notified and a QC status code is returned to the application. (MPPs).

A batch message processing region (BMP) scheduled against wait-for-input (WFI) transactions returns a QC status code (no more messages) for /PSTOP REGION, /DBD, /DBR, or /STA commands only.

Examples

The following is an example of the /STOP CLASS command:

Entry ET:

```
/STOP CLASS 3
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: No further scheduling of application programs for class 3 transactions occurs. All message processing programs currently handling class 3 transactions are allowed to run until the processing limit count is reached or the input queue contains no more messages.

/STOP DATAGRP command

The /STOP DATAGRP command specifies groups of DL/I databases, Fast Path DEDBs, and Fast Path areas to be stopped.

Data groups are logical groupings of databases and areas; they enable simplified command processing for databases and areas. You define a data group in the RECON data set by using the INIT.DBDSGRP command with parameters GRPNAME and DBGRP. DATAGRP is not valid on RSR tracking subsystems.

Subsections:

- "Environment"
- "Syntax" on page 516
- "Usage notes" on page 516
- "Equivalent IMS type-2 commands" on page 516

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 245. Valid environments for the /STOP DATAGRP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DATAGRP	X	X	
LOCAL	X	X	

Syntax

Usage notes

During /STOP DATAGRP processing, all virtual storage option (VSO) DEDBs that are in a z/OS data space are removed from the data space and updates are written out to DASD.

Recommendation: Although you can use DBDS groups as well as DB groups for this command, you should use DB groups whenever possible to eliminate the overhead of converting the DBDS group to a DB group.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP DATAGRP command and the IMS type-2 commands that perform similar functions.

Table 246. Type-2 equivalents for the /STOP DATAGRP command

Task	/STOP DATAGRP command	Similar IMS type-2 command
Stops a data group.	/STOP DATAGRP *datagrpname*	UPDATE DATAGRP NAME(*datagrpname*) STOP(SCHD)

Related reference:

"UPDATE DATAGRP command" on page 643

/STOP DB command

The /STOP DB command prevents subsequently scheduled programs from accessing the database, without affecting currently scheduled programs or closing the database.

Subsections:

- "Environment"
- "Syntax" on page 517
- "Usage notes" on page 517
- "Equivalent IMS type-2 commands" on page 519
- "Examples" on page 519

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

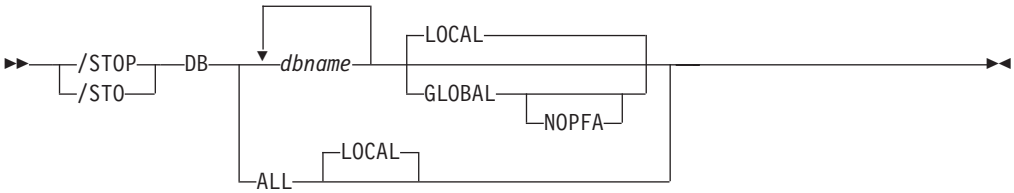
Table 247. Valid environments for the /STOP DB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DB	X	X	
GLOBAL	X	X	

Table 247. Valid environments for the /STOP DB command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
LOCAL	X	X	
NOPFA	X	X	

Syntax



Usage notes

If the database is a DEDB or MSDB, programs using the database will not be scheduled. For other databases, the programs will still be scheduled but a call against the database will result in either a 3303 pseudoabend, or a BA status code, if the INIT call was issued.

If the database is stopped after the region is scheduled, the region is not allowed to wait for the next message (wait-for-input mode). If there are no more messages available for the region to process, a QC status (no more messages) will be returned to the application (MPPs). If the region is already scheduled and waiting for the next message (wait-for-input mode) when the command is entered, the region is notified and a QC status code is returned to the application (MPPs).

A batch message processing region (BMP) scheduled against wait-for-input (WFI) transactions returns a QC status code (no more messages) for /PSTOP REGION, /DBD, /DBR, or /STA commands only.

In an IFP region, the /STOP command has no effect until the region is started again. Issuing a /STOP DB command with the ACCESS parameter is not valid for an MSBD.

The GLOBAL command is processed by the IMS system where the command was initiated. This system will process the command locally and then request IRLM NOTIFY to route and process the command on sharing IMS systems.

If global DB status is maintained, the global status maintained in RM is also updated. The global status is set to STOSCHD.

If the command is entered from OM API, the global status is updated by the command master IMS. If the command is not entered from OM API, the IMS that initiated the GLOBAL command updates the global status in RM.

If global status in RM is successfully updated, message DFS0988I for RSRCTYPE=DB is issued. If global status is not successfully updated, message DFS3308I is issued, indicating RM failure, and no command response lines are

generated. Any RM error is traced to the OCMD trace table. Users can issue a QRY DB STATUS(GLOBAL) command to obtain the global status of the resources in RM.

The X'4C' log record for databases is updated to include both global status and global command time stamp.

For DBCTL, when CCTL schedules a PSB, the DBCTL thread SCHED request defines the thread as LONG or SHORT. If the database is currently scheduled to a LONG thread, the command is rejected; otherwise, the thread is allowed to complete before the database is acted upon. This results in either a commit point or transaction termination.

The output of the/STOP DB command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate to the database type and the command completion.

- Full-function database messages: DFS132, DFS160, DFS216, DFS0488I, DFS1407, DFS2026, DFS3318I, DFS3466I
- Fast Path database messages: No unique messages are returned.

See the AREA keyword for a description of the LOCAL and NOPFA keywords.

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB except for the /START DB command and the UPDATE DB START(ACCESS) command. A HALDB partition is not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

The HALDB partition reflects conditions such as STOPPED (shown as STOACC or STOSCHD or STOUPDS on QUERY DB), LOCKED, or NOTOPEN. When a HALDB partition is stopped, it must be explicitly started again. The type-1 commands with the keyword ALL, type-2 commands with NAME(*), and commands against a HALDB master do not change the STOPPED (shown as STOACC, STOSCHD or STOUPDS on QUERY DB) and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is /DBR on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. On a /DISPLAY DB command, only the HALDB master displays STOPPED (each HALDB partition does not display STOPPED unless it was itself stopped). If a /DBR command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

Each partition inherits the access limitations of its HALDB master. If the /DBD command is issued against a HALDB master, all of its partitions close. A subsequent reference to any of the partitions results in the partition opening for input, although the partition's access might be UPDATE or EXCLUSIVE. The DBRC authorization state reflects the limited access.

Restrictions:

1. The /STOP DB command is not supported on an RSR tracking subsystem.
2. The /STOP DB command cannot be processed against a HALDB partition on an IMS system while HALDB Online Reorganization (OLR) is running against that partition on the same IMS system.
3. While the database is being quiesced, this command cannot be processed successfully.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP DB command and the IMS type-2 commands that perform similar functions.

Table 248. Type-2 equivalents for the /STOP DB command

Task	/STOP DB command	Similar IMS type-2 command
Stops a database.	/STOP DB *dbname*	UPDATE DB NAME(*dbname*) STOP(SCHD)

Examples

The following are examples of the /STOP DB command:

Example 1 for /STOP DB command

Entry ET:

```
/STOP DATABASE TREEFARM
```

Response ET:

```
DFS058I  STOP COMMAND IN PROGRESS
DFS0488I  STOP COMMAND COMPLETED.  DBN=TREEFARM RC=0
```

Explanation: Database TREEFARM is stopped.

Example 2 for /STOP DB command

TSO SPOC input:

```
STO DB BANKATMS BANKTERM BANKLDGR BE3ORDER
```

TSO SPOC output:

```
SYS3     DFS0488I  STO COMMAND COMPLETED. DBN= BANKATMS RC= 0
SYS3     DFS0488I  STO COMMAND COMPLETED. DBN= BANKTERM RC= 0
SYS3     DFS0488I  STO COMMAND COMPLETED. DBN= BANKLDGR RC= 0
SYS3     DFS0488I  STO COMMAND COMPLETED. DBN= BE3ORDER RC= 0
IMS3     DFS0488I  STO COMMAND COMPLETED. DBN= BANKATMS RC= 0
IMS3     DFS0488I  STO COMMAND COMPLETED. DBN= BANKTERM RC= 0
IMS3     DFS0488I  STO COMMAND COMPLETED. DBN= BANKLDGR RC= 0
IMS3     DFS0488I  STO COMMAND COMPLETED. DBN= BE3ORDER RC= 0
```

OM API input:

```
CMD (  STO DB BANKATMS BANKTERM BANKLDGR BE3ORDER )
```

OM API output:

```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
```

```
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.197 22:05:21.270547</statime>
<stotime>2002.197 22:05:21.307712</stotime>
<staseq>B7EFC16AF6B13F26</staseq>
<stoseq>B7EFC16AFFC40D8C</stoseq>
<rqsttkn1>USRT005 10150521</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT005 </userid>
<verb>STO </verb>
<kwd>DB      </kwd>
<input>STO DB BANKATMS BANKTERM BANKLDGR BE3ORDER  </input>
</cmd>
<msgdata>
<mbr name="SYS3    ">
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKATMS RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKTERM RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKLDGR RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BE3ORDER RC= 0</msg>
</mbr>
<mbr name="IMS3    ">
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKATMS RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKTERM RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKLDGR RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BE3ORDER RC= 0</msg>
</mbr>
</msgdata>
</imsout>
```

Explanation: The STOP command is routed from OM to the two active IMS systems - SYS3 and IMS3. The response from both IMS systems is returned to OM. The databases BANKATMS, BANKTERM, BANKLDGR, and BE3ORDER are stopped at both IMS systems.

Related concepts:

Maintaining global information for databases, DEDB areas, and transactions (System Administration)

Related reference:

"UPDATE DB command" on page 660

/STOP DC command

The /STOP DC command prohibits you from logging on to VTAM and ensures that all VTAM node sessions have terminated before IMS issues the DFS2111I message, which says the ACB is closed.

Subsections:

- "Environment"
- "Syntax" on page 521
- "Usage notes" on page 521

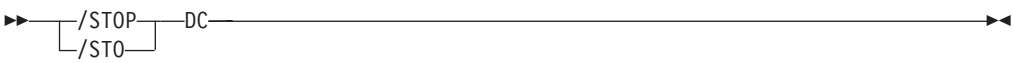
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 249. Valid environments for the /STOP DC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DC	X		X

Syntax



Usage notes

There are two ACBs if MNPS for XRF is used. If you use XRF with MNPS, both the APPLID and MNPS ACB are closed. If you are not using XRF or using XRF without MNPS, there is only one ACB, the VTAM ACB, which is closed. APPLID ACB is the same as VTAM ACB.

The /STOP DC command can be used either before or after the /CLSDST NODE or /STOP NODE command, the only difference being that logons can still occur if the /STOP DC command is not entered. However, the command cannot start or complete processing if the VTAM ACBs (APPLID and MNPS) are not open or the VTAM nodes remain active. If the nodes are active, the /CLSDST NODE or /STOP NODE command must be issued to close the nodes; in some cases, a /IDLE NODE command can be issued to cause an OS VTAM VARY command to be issued against any nodes that remain connected.

/STOP DESC command

The /STOP DESC command defines the LU62 descriptors from DFS62DTx PROCLIB member to IMS.

Subsections:

- "Environment"
- "Syntax"

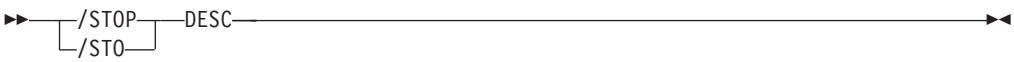
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 250. Valid environments for the /STOP DESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DESC	X		X

Syntax



/STOP LINE command

The /STOP LINE command stops message queuing for lines and stops the sending and receiving of messages over the lines. However, lines are not considered stopped unless they are stopped and idle. Use /DISPLAY LINE to verify line status.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples"

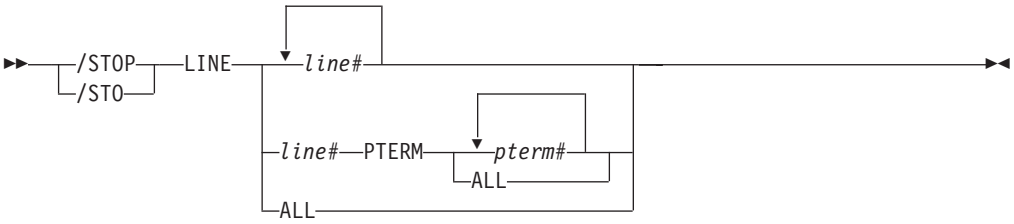
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 251. Valid environments for the /STOP LINE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
LINE	X		X
PTERM	X		X

Syntax



Usage notes

The /STOP LINE PTERM command ensures that no input messages from any of the specified terminals assigned to the specified lines will be received by IMS after the command is issued.

The /STOP LINE command resets preset mode, test mode, response mode, lock pterm, lock lterm, pstop lterm, and purge lterm because these statuses are not significant and therefore are not kept after a /START LINE or restart.

Examples

The following are examples of the /STOP LINE command:

Example 1

Entry ET:

```
/STOP LINE 4,5,6,7,8,9,10,11
```

Response ET:

```
  DFS058I  STOP COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL STOPPED
```

Explanation: Lines 4, 5, 6, 7, 8, 9, 10, and 11 and their associated physical terminals are stopped.

Example 2

Entry ET:

```
/STOP LINE 4 8 900
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED EXCEPT LINE 900
```

Response RT:

```
DFS059I  TERMINAL STOPPED
```

Explanation: Lines 4 and 8 and their associated physical terminals are stopped. 900 is an invalid line number.

Example 3

Entry ET:

```
/STOP LINE 4 PTERM 1, 2
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL STOPPED
```

Explanation: Physical terminals 1 and 2 on line 4 are stopped.

/STOP LTERM command

The /STOP LTERM command specifies the LTERM that is to be stopped. The /STOP LTERM command with a logical terminal that is in a QLOCKED state does not reset the QLOCK state, but puts the LTERM in a STOPPED and QLOCKED state.

Subsections:

- "Environment" on page 524
- "Syntax" on page 524
- "Usage notes" on page 524
- "Examples" on page 524

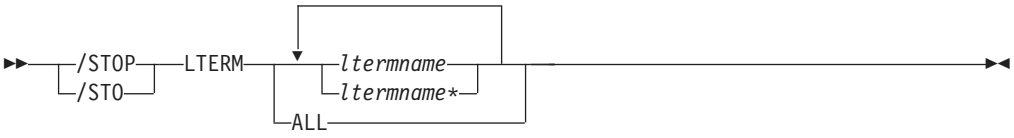
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 252. Valid environments for the /STOP LTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
LTERM	X		X

Syntax



Usage notes

If IMS internally resets the QLOCK condition, the LTERM remains in a STOPPED state. (QLOCK indicates that the LTERM is locked from sending any further output or from receiving input that can create additional output for the same LTERM until the state is reset by a specific request received on the session.)

The /STOP LTERM command is rejected for remote logical terminals.

The LTERM parameter can be generic where the generic parameter specifies LTERMs that already exist.

If global resource information is kept in Resource Manager, the /STOP LTERM command stops messages from being queued to the lterm anywhere in the IMSplex and the change is reflected both in Resource Manager and in the local IMS system.

Examples

The following is an example of the /STOP LTERM command:

Entry ET:

```
/STOP LTERM APPLE, TREE, FRUIT
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL STOPPED
```

Explanation: Logical terminals APPLE, TREE, and FRUIT are stopped.

/STOP LUNAME command

The /STOP LUNAME command specifies a particular LU name that is to be stopped.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"

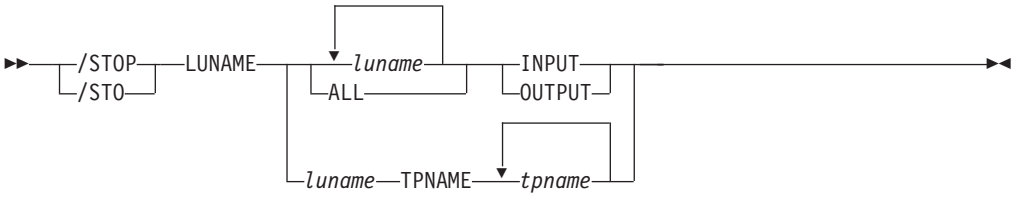
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 253. Valid environments for the /STOP LUNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
INPUT	X		X
LUNAME	X		X
OUTPUT	X		X

Syntax



Usage notes

Specifying the keyword INPUT with the LUNAME keyword stops an LU name for any input and synchronous outbound activities. Specifying the parameter ALL with INPUT causes all future LU 6.2 input and synchronous outbound activities to be stopped as well.

Specifying the keyword OUTPUT with the LUNAME keyword stops an LU name for any asynchronous outbound activities. Specifying the parameter ALL with OUTPUT causes all future LU 6.2 asynchronous outbound activities to be stopped as well.

Specifying neither INPUT nor OUTPUT is the same as specifying both INPUT and OUTPUT. The LU name is stopped for any input, and both synchronous and asynchronous outbound activities. Specifying the parameter ALL in this case stops all future LU 6.2 inbound activities, synchronous and asynchronous outbound activities.

A network-qualified LU name is optional for the LUNAME keyword. If the LU name is not network-qualified and no TP name is specified, all network-qualified LU names whose LU names match the LU name specified are also stopped.

The /STOP LUNAME TPNAME command stops a particular TP name of the LU name specified. The keyword OUTPUT is the default for this command.

If the specified resource does not exist, a structure is created to retain the status.

Related reference:

Command keywords and their synonyms (Commands)

/STOP MADSIOT command

The /STOP MADSIOT command allows users to disable the MADS I/O timing function in a MADS I/O timing enabled environment.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"

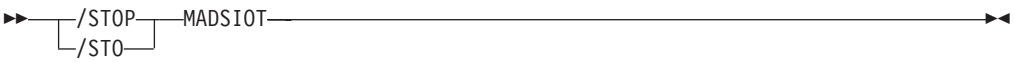
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 254. Valid environments for the /STOP MADSIOT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
MADSIOT	X	X	

Syntax



Usage notes

When /STOP MADSIOT completes normally, the following message is returned to the operator's console:

```
DFS12761 MADS I/O TIMING FUNCTION STOPPED SUCCESSFULLY
```

/START MADSIOT allows users to resume the MADS I/O timing function.

If MADS I/O Timing list structure is not defined in DFSVSMxx, the command will be rejected. If MADS I/O Timing function is already disabled, the command will be ignored. If MADS I/O Timing function is enabled and all sharing partners successfully disconnect from MADS I/O Timing list structure on the coupling facility, the command will complete successfully; if any sharing partners fails to disconnect to MASD I/O Timing list structure, the command will fail.

The output of the/STOP MADSIOT command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS0023I, DFS0008I, DFS1271I, DFS1276I, DFS1275E, DFS1219E

/STOP MSNAME command

The /STOP MSNAME command stops the sending of all messages (primary requests) from a terminal except those continuing a conversation. This includes all messages destined for remote transactions with the SYSID of the MSNAME and for remote logical terminals associated with this MSNAME.

Subsections:

- "Environment"
- "Syntax"
- "Examples"

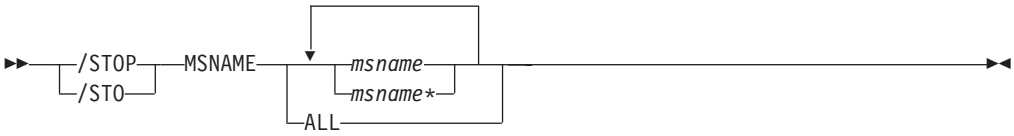
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 255. Valid environments for the /STOP MSNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
MSNAME	X		X

Syntax



Examples

The following is an example of the /STOP MSNAME command:

Entry ET:

```
/STOP MSNAME BOSTON
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: The logical link path associated with the name BOSTON is stopped.

Related reference:

"UPDATE MSNAME command" on page 717

/STOP NODE command

The /STOP NODE command specifies the VTAM node to be stopped and logged off. The NODE parameter can be generic if the USER keyword is not specified and applies to nodes that already exist.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 529

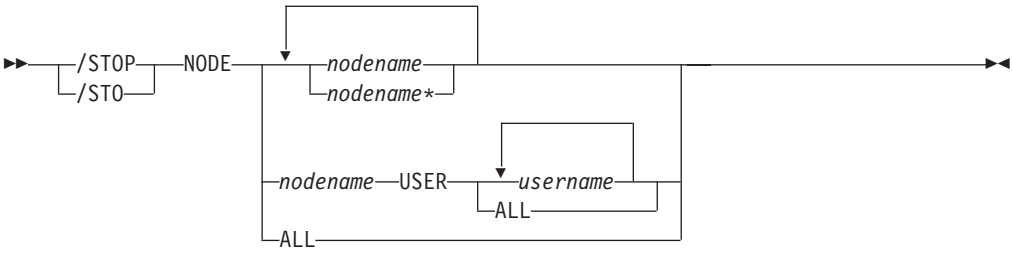
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 256. Valid environments for the /STOP NODE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
NODE	X		X
USER	X		X

Syntax



Usage notes

The /STOP NODE command prevents future logons until a /START NODE command is issued.

/STOP NODE without the USER keyword is supported for nodes that do not yet exist. It causes the node to be created and stopped which prevents the dynamic node from logging on. /STOP NODE without the USER keyword affects all half-sessions of the specified node.

/STOP NODE USER is valid for ISC and non-ISC nodes and users; however the user must still be allocated or signed on to the node. /STOP NODE USER for ISC nodes stops the named half-session defined in USER username for NODE nodename.

Restrictions for using NODE and USER parameters together:

- Commands with the NODE USER keyword pair are valid only if:

  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- /STOP NODE USER commands are valid for ISC and non-ISC nodes and users.

/STOP NODE resets preset mode, test mode, lock node, lock lterm, pstop lterm, and purge lterm, because these statuses are not significant and therefore are not kept after a logon or restart. For Fast Path input response mode, you must also issue the /START NODE command before the mode is reset. The /STOP NODE command also takes other actions depending on the recovery settings for the node:

RCVYSTSN=NO
: /STOP NODE acts like a /CHANGE NODE COLDSESS command for FINANCE and SLUP nodes by setting the session status to 'cold'. /STOP NODE acts like a /QUIESCE NODE command for ISC (LU6.1) nodes by initiating the shutdown and deallocating the user for the specified node. This action changes the session status to 'cold'. With these actions taken by the /STOP NODE command, the next session initiation request for this node is allowed to again attempt a session cold start (after a /START NODE command has been entered).

RCVYCONV=NO
: /STOP NODE causes any IMS conversations (active and held) to be terminated. Any conversational message that is queued or being processed will have its output response message delivered asynchronously.

RCVYFP=NO
: /STOP NODE causes Fast Path status and messages to be discarded

RCVYRESP=NO
: /STOP NODE resets full-function response mode.

If global resource information is kept in Resource Manager, the /STOP NODE command sets a global stop status for the node and prevents the node from logging on anywhere in the IMSplex. If global resource information is not kept in Resource Manager, /STOP NODE creates the node, if it does not exist in an ETO environment, and sets stop status for the local node. If the node does not exist in a non-ETO environment, the /STOP NODE command is rejected.

Examples

The following is an example of the /STOP NODE command:

Entry ET:

```
/STOP NODE HARRY
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: The physical terminal associated with node HARRY is disconnected (/CLSDST) and further logons are prevented.

/STOP OLDS command

The /STOP OLDS command indicates that IMS is to stop using an OLDS log data set.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Examples"

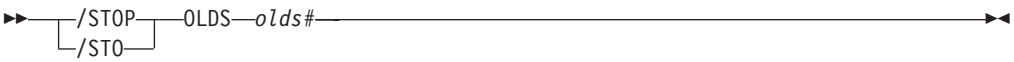
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 257. Valid environments for the /STOP OLDS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
OLDS	X	X	X

Syntax



Keywords

The following keyword is valid for the /STOP OLDS command:

olds#
: Identifies an OLDS that is defined by JCL or a DFSMDA macro and is currently started. *olds#* must be 00 through 99.

Usage notes

The stopped OLDS will be dynamically deallocated when it is no longer possible for it to be accessed for dynamic backout.

If in dual mode, both primary and secondary OLDSs are stopped. If there are only two OLDS data sets available, or if the specified OLDS is the one currently being used for output, the /STOP OLDS command will be rejected.

Examples

The following is an example of the /STOP OLDS command:

Entry ET:

```
/STOP OLDS 09
```

Response ET:

```
DFS058I  STOP COMMAND IN PROGRESS
DFS2500I DATASET DFSOLP09 SUCCESSFULLY DEALLOCATED
DFS3257I OLDS DEALLOCATED ON DFSOLP09
```

Explanation: The subject OLDS, DFSOLP09 (DFSOLS09), will be stopped.

/STOP OTMA command

The /STOP OTMA command causes IMS to leave the z/OS cross-system coupling facility (XCF) group for IMS Open Transaction Manager Access (OTMA).

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 258. Valid environments for the /STOP OTMA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
OTMA	X		X

Syntax

Usage notes

/STOP OTMA command processing is as follows:

1. IMS leaves the XCF group.
2. For any IMS OTMA output awaiting an ACK message, IMS aborts the message. For Commit-then-Send transactions, the output remains enqueued to the transaction pipe. For Send-then-Commit transactions, IMS aborts the transaction.

When the /STOP OTMA command is issued, it will clear or reject all the ICAL messages for all the transaction pipes.

Examples

The following is an example of the /STOP OTMA command:

Entry ET:

```
/STO OTMA
```

Response ET:

```
DFS2361I 14:02:05 XCF GROUP CLOSED SUCCESSFULLY. SYS3
DFS058I 14:02:06 STOP COMMAND COMPLETED   SYS3
DFS996I *IMS READY*  SYS3
```

/STOP PGM command

The /STOP PGM command specifies the application program that is to be stopped.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 259. Valid environments for the /STOP PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
PGM	X	X	X

Syntax

Usage notes

The /STOP PGM command does not stop CPI Communications driven transaction programs.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP PGM command and the IMS type-2 commands that perform similar functions.

Table 260. Type-2 equivalents for the /STOP PGM command

Task	/STOP PGM command	Similar IMS type-2 command
Stops program scheduling.	/STOP PGM *pgmname*	UPDATE PGM NAME(*pgmname*) STOP(SCHD)

Examples

The following is an example of the /STOP PGM command:

Entry ET:

```
/STOP PROGRAM APPLETRE
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: Application program APPLETRE is stopped.

Related reference:

"UPDATE PGM command" on page 755

/STOP REGION command

Use the /STOP REGION command to stop IMS regions, application programs, or both. /STOP REGION is not mirrored on the XRF alternate system. You must enter this command on the alternate system if you want it to affect the alternate system.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 534
- "Examples" on page 535

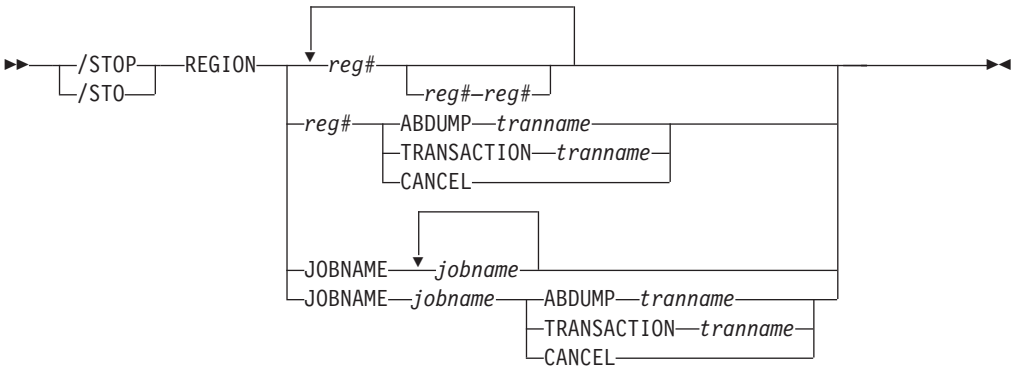
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 261. Valid environments for the /STOP REGION command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
ABDUMP	X	X	X
CANCEL	X	X	X
JOBNAME	X	X	X
REGION	X	X	X
TRANSACTION	X		X

Syntax



Keywords

The following keywords are valid for the /STOP REGION command:

REGION *reg#*
Is used to terminate one or more message processing regions at the conclusion of processing the current transaction. The region identifier is *reg*#.

/STOP REGION *reg*# can also be used to terminate Fast Path regions. /STOP REGION *reg*# cannot be used to terminate batch regions.

A Fast Path utility region is terminated at the next system checkpoint of the utility.

REGION *reg#–reg#*
Is used to terminate a range of message processing regions at the conclusion of processing the current transaction.

REGION *reg#* **ABDUMP** *tranname*
Causes abnormal termination of an application program.

The region identifier is *reg*# and the transaction code is *tranname*.

If the transaction indicated by *tranname* is currently running in REGION *reg*#, an error message is received at the master terminal, indicating an application program abend. The region will remain active, but the transaction will be stopped. The command is ignored if the transaction is not currently scheduled in region *reg*#.

/STOP REGION *reg*# ABDUMP should be used only for a region that appears to be looping or in a wait state. If this command does not abnormally terminate the application running in a region, the /STOP REGION *reg*# CANCEL command can be used. This might cause the control region to terminate with abend 113 if parallel DL/I is being used. See /STOP REGION *reg*# CANCEL for further warnings.

If the /CHECKPOINT command cannot shut down IMS because a message processing region appears to be active, but the region is no longer active in the system (a condition commonly referred to as a phantom region), the /STOP REGION *reg*# ABDUMP command can be used to correct the situation. In this case, the /STOP REGION *reg*# ABDUMP command detects that the region is no longer active and cleans the internal IMS entries for the nonexistent region, which allows the shut down process to proceed normally.

If a message processing region experiences a catastrophic failure and abnormally terminates and a /DISPLAY ACTIVE REGION shows the region is still defined to IMS, the /STOP REGION *reg*# ABDUMP command can be used to correct the situation. In this case, the /STOP REGION *reg*# ABDUMP command detects the region is no longer active and cleans the internal IMS entries for the nonexistent region.

The *tranname* variable is not valid for batch, IFP, or Fast Path utility regions.

REGION *reg#* **TRANSACTION** *tranname*
Stops a message processing program in wait-for-input (WFI) mode from processing within the specified region.

The region identifier is *reg*# and the transaction code is *tranname*.

If the transaction indicated by *tranname* is currently running in region *reg*#, the IMS message DFS0569I is received at the master terminal, indicating that a QC status code (no more messages) was returned to the application program (MPPs). The region that contained the application is still active and the

transaction is not stopped. A batch message processing program in WFI mode must be stopped using the /PSTOP command.

A batch message processing region (BMP) scheduled against wait-for-input (WFI) transactions returns a QC status code (no more messages) for, /PSTOP REGION, /DBD, /DBR, or /STA commands only.

REGION *reg#* **CANCEL**
Is used if the region cannot be stopped with a /STOP REGION ABDUMP command and must be preceded by a /STOP REGION ABDUMP command.

The region identifier is *reg*#.

Using the /STOP REGION CANCEL command can cause the IMS control region to terminate with user abend 113 if parallel DL/I is being used. A z/OS CANCEL command will be rejected.

REGION JOBNAME
Identifies regions to be stopped by their job names. The job name must be 1-8 alphanumeric or national ($,#,@) characters. The first character of the job name must be either alphabetic or national.

Examples

The following are examples of the /STOP REGION command:

Example 1 for /STOP REGION command

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24                  1,  2
    1 BMP       BMP   BMP       BMP255
    3 IFPN      FPM   NO MSG.   DDLTJN26
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/142004*
```

Explanation: Fast Path message-driven region 3 currently has no messages to process.

Entry ET:

```
/STOP REG 3
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
```

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME    TYPE  TRAN/STEP PROGRAM  STATUS         CLASS
    2 MPP        TP    TXCDRN24  DDLTRN24                 1,   2
    1 BMP        BMP   BMP       BMP255
```

```
      FPRGN      FP    NONE
      DBRC1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -    1 ACTIV-OUT -   0
NODE ACTIVE-IN -    0 ACTIV-OUT -   0
LINK ACTIVE-In -    0 ACTIV-OUT -   0
*89041/142102*
```

Explanation: Fast Path region 3 has been stopped. If region 3 had been processing a message, IMS would have terminated the region on completion of the transaction.

Example 2 for /STOP REGION command

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS         CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24                 1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/142102*
```

Entry ET:

```
/STOP REG 1
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS0557I STOP REGION ID NOT VALID- REGION 0001 IS BMP.
```

Explanation: /STOP REGION (with no keywords) is not valid for batch regions.

Example 3 for /STOP REGION command

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24                   1,   2
    1 BMP       BMP   BMP       BMP255
    3 FPU       FPU   IFP       DBF#FPU0
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/142453*
```

Entry ET:

```
/STOP REG 3
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
```

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24                  1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-In -   0 ACTIV-OUT -   0
*89041/142758*
```

Explanation: Fast Path utility region 3 has been stopped. The Fast Path utility DBF#FPU0 was terminated at the next system checkpoint.

Example 4 for /STOP REGION command

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24                  1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE   SYS3
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/142758*
```

Explanation: Transaction TXCDRN24 in region 2 is looping or in a wait state.

Entry ET:

```
/STOP REG 2 ABDUMP TXCDRN24
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS555I TRAN TXCDRN24 ABEND S000,U0474 SYS ID 220  MSG IN PROGRESS
```

Explanation: The application program has been terminated with a U0474 ABEND. This abend indicates termination in response to a user request (/STOP REGION ABDUMP).

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS         CLASS
    2 MPP       TP    WAITING                           1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
```

```
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/143420*
```

Explanation: The application has been terminated but the region remains active.

Entry ET:

```
/DISPLAY PROG DDLTRN24
```

Response ET:

```
PROGRAM   TRAN       TYPE
DDLTRN24   TXCDRN24   TP
*90340/143749*
```

Explanation: The program has not been stopped.

Entry ET:

```
/DISPLAY TRANSACTION TXCDRN24
```

Response ET:

```
  TRAN   CLS ENQCT   QCT   LCT  PLCT CP NP LP SEGSZ SEGNO PARLM RC
TXCDRN24   2     1     0 65535 65535  1  1  1     0     0     0  0
PSBNAME: DDLTRN24
STATUS: STOP
*90340/143802*
```

Explanation: The transaction has been stopped.

Example 5 for /STOP REGION command

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    2 MPP       TP    WAITING                              1,   2
    1 BMP       BMP   BMP       BMP255
    3 FPU       FPU   IFP       DBP#FPU0
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/144248*
```

Entry ET:

```
/STOP REG 3 ABDUMP
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
```

Explanation: A transaction code is not entered when terminating a Fast Path utility with a /STOP REGION ABDUMP command.

Example 6 for /STOP REGION command

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS            CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24 WAIT-INPUT          1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINE ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/150141*
```

Explanation: Message processing program DDLTRN24 is waiting for an input message.

Entry ET:

```
/STOP REGION 2 TRANSACTION TXCDRN24
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS0569I PSTOP OR STOP COMPLETE FOR REGION0002 TRAN TXCDRN24.
```

Explanation: A QC status code was returned to the WFI application program DDLTRN24.

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS         CLASS
    2 MPP       TP    WAITING                           1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-In -   0 ACTIV-OUT -   0
*89041/150206*
```

Explanation: The WFI application has been terminated but the region is still active.

Entry ET:

```
/DISPLAY TRANSACTION TXCDRN24
```

Response ET:

```
  TRAN   CLS ENQCT   QCT   LCT  PLCT CP NP LP SEGSZ SEGNO PARLM RC
TXCDRN24   2     4     0 65535 65535  1  1  1     0     0     0  0
PSBNAME: DDLTRN24
*90340/150219*
```

Explanation: The transaction is not stopped.

Example 7 for /STOP REGION command

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24 WAIT-INPUT         1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/150813*
```

Entry ET:

```
/STOP REGION 2 TRANSACTION TRAN255
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS0558I TRAN TRAN255 NOT SCHEDULED
```

Explanation: TRAN255 is a valid transaction for the IMS system but it is not currently scheduled in region 2. If TRAN255 had not been a valid transaction for the IMS system, only message DFS230I (TRAN SPECIFIED WITH ABDUMP OR TRAN KEYWORD IS NOT VALID) would have been issued.

Example 8 for /STOP REGION command

Entry ET:

```
 D A,L
```

Response ET:

```
 JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
 00001    00010    00001      00019    00020    00001/00020       00000
 LLA      LLA      LLA      NSW  S  VLF      VLF      VLF      NSW  S
 JES2     JES2     IEFPROC  NSW  S  RMF      RMF      IEFPROC  NSW  S
 IMSVTAM  IMSVTAM  IEFPROC  NSW  S  TSO      TSO      STEP1    NSW  S
 CQS      CQS      IEFPROC  NSW  S  IMSECTA9 IMSECTA9 IEFPROC  NSW  S
 DLIECTA9 DLIECTA9 DLISAS   NSW  S  DBRECTA9 DBRECTA9 DBRC     NSW  S
 MPP610C  MPP      MPP      NSW  J
 USRT001  OWT
```

Explanation: MPP610C is an IMS message processing region.

Entry ET:

```
/STOP REGION JOBNAME MPP610C
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS552I MESSAGE REGION MPP610C  STOPPED ID=00001 TIME=1616  SYSX

SMF000I  MPP610C     MPP         DFSRRC00    0000
$HASP395 MPP610C  ENDED
```

Related reference:

"/STOP THREAD command" on page 546

/STOP RTC command

Use the /STOP RTC command to specify that transactions associated with this routing code are not processed.

Subsections:

- "Environment"
- "Syntax"
- "Equivalent IMS type-2 commands"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 262. Valid environments for the /STOP RTC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
RTC	X		X

Syntax

Equivalent IMS type-2 commands

The following table shows variations of the /STOP RTC command and the IMS type-2 commands that perform similar functions.

Table 263. Type-2 equivalents for the /STOP RTC command

Task	/STOP RTC command	Similar IMS type-2 command
Stops queuing to a Fast Path routing code.	/STOP RTC *rtcname*	UPDATE RTC NAME(*rtcname*) STOP(Q)

Related reference:

"UPDATE RTC command" on page 782

/STOP SB command

Use the /STOP SB command to disallow further use of sequential buffering. /STOP SB does not affect sequential buffering applications scheduled before this command was issued.

Subsections:

- "Environment" on page 542
- "Syntax" on page 542
- "Examples" on page 542

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 264. Valid environments for the /STOP SB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SB	X	X	

Syntax

Examples

The following is an example of the /STOP SB command:

Entry ET:

```
/STOP SB
```

Response ET:

```
DFS058  STOP COMMAND COMPLETED
```

Entry ET:

```
/DISPLAY POOL DBAS
```

Response ET:

```
SEQUENTIAL BUFFERING:  STATUS = STOPPED
  MAX      N.A.  FREE   N.A.   CURR     0K  HIGH  320K
DATABASE BUFFER POOL: SIZE   67584
  REQ1       0 REQ2     0 READ     0 BISAM     0 WRITES     0
  KEYC       0 LCYL     0 PURG     0 OWNRR     0 ERRORS 00/00
  DATABASE BUFFER POOL:  BSIZE  12288
    RRBA     0 RKEY     0  BFALT    0 NREC  0 SYN PTS     0
    NMBUFS  29 VRDS     0   FOUND   0 VWTS  0  ERRORS 00/00
  DATABASE BUFFER POOL:  BSIZE  356352
    RRBA     0 RKEY     0  BFALT    0 NREC  0 SYN PTS     0
    NMBUFS  29 VRDS     0  FOUND    0 VWTS  0  ERRORS 00/00
  *90253/104547*
```

/STOP SERVGRP command

Use the /STOP SERVGRP command to stop communications between the service group in an RSR complex at which the command was entered and the service group at the other site. /STOP SERVGRP also severs the relationship between the IMS subsystem and the TMS subsystem.

Subsections:

- "Environment" on page 543
- "Syntax" on page 543
- "Usage notes" on page 543

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 265. Valid environments for the /STOP SERVGRP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SERVGRP	X	X	X

Syntax

Usage notes

Once communications are stopped, the logger stops sending log data to the RSR tracking subsystem. No more attempts to re-establish failed conversations are made at OLDS switch. The /STOP SERVGRP command is normally not needed. The /STOP SERVGRP command is valid from an active subsystem and a tracking subsystem.

Successful completion of the syntax checking of the /STOP SERVGRP command results in the `DFS058 STOP COMMAND COMPLETED` message, although processing of the command continues asynchronously.

/STOP SLDSREAD command

The /STOP SLDSREAD command indicates whether IMS is enabled to retrieve records from both a system log data set (SLDS) and OLDS or OLDS only. The default is that SLDSREAD is enabled.

Subsections:

- "Environment"
- "Syntax" on page 544
- "Usage notes" on page 544

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 266. Valid environments for the /STOP SLDSREAD command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SLDSREAD	X	X	X

Syntax

Usage notes

If the SLDSREAD process is active, issuing the /STOP SLDSREAD command causes a U4095 symptom dump to be taken, and the SLDSREAD processing is stopped. This abend is not fatal, and IMS continues to function normally. Any backout processes that were active have to be backed out manually by using the batch backout utilities.

If the SLDSREAD process is not active, issuing the /STOP SLDSREAD command prevents any SLDSREAD processes from being started if an SLDS should be needed for backout processing. Use the batch backout utilities to back out any application programs that failed.

/STOP SUBSYS command

Use the /STOP SUBSYS command to specify the name of the external subsystem whose connection is to be terminated.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 545

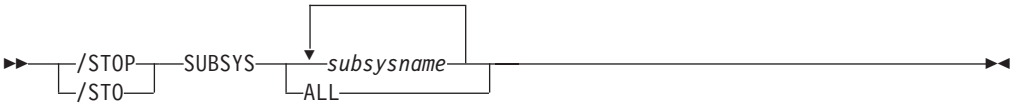
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 267. Valid environments for the /STOP SUBSYS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SUBSYS	X	X	X

Syntax



Usage notes

The /STOP SUBSYS command does enable application programs currently accessing external resources to complete normally. When those applications have terminated, the connection to the subsystem will also terminate. The application must complete all message processing before actual connection termination. The

next occurrence of an external subsystem call will receive a nonzero return code, indicating the connection is not available. A /START command is then necessary to reestablish the connection.

The /STOP SUBSYS command can also be used to dynamically reconfigure existing subsystem definitions. The operator can issue the /STOP SUBSYS command, change or add to the PROCLIB member, and then issue the /START SUBSYS command. IMS attempts to connect those subsystems defined in the PROCLIB member.

If system failure occurs after the /STOP SUBSYS command is processed, the stopped status is still set.

Examples

The following are examples of the /STOP SUBSYS command:

Example 1

Entry ET:

```
/STOP SUBSYS ALL
```

Response ET:

```
DFS058I  STOP COMMAND IN PROGRESS
```

Explanation: IMS has initiated the termination of the connection. When all dependent regions have terminated their connections, IMS will complete the termination. It is likely that an external subsystem message indicating connection termination will be received at this time.

Example 2

Entry ET:

```
/STOP SUBSYS XXX1 XXX3
```

Response ET:

```
DFS058I  STOP COMMAND IN PROGRESS
```

Explanation: IMS has initiated the termination of the connection. When all dependent regions have terminated their connections, IMS will complete the termination. It is likely that an external subsystem message indicating connection termination will be received at this time

/STOP SURV command

Use the /STOP SURV command in an XRF environment to stop the operation of the IMS surveillance function.

Subsections:

- "Environment" on page 546
- "Syntax" on page 546
- "Keywords" on page 546

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 268. Valid environments for the /STOP SURV command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SURV	X		X

Syntax

Keywords

The following keywords are valid for the /STOP SURV command:

ALL
The same as specifying LNK, LOG, and RDS. This is the default.

LNK
ISC link.

LOG
System log.

RDS
Restart data set.

/STOP THREAD command

Use the /STOP THREAD command to stop an inactive CCTL thread. The DEDB utility region is terminated at the next system checkpoint.

Subsections:

- "Environment"
- "Syntax" on page 547
- "Keywords" on page 547
- "Usage notes" on page 547
- "Examples" on page 547

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

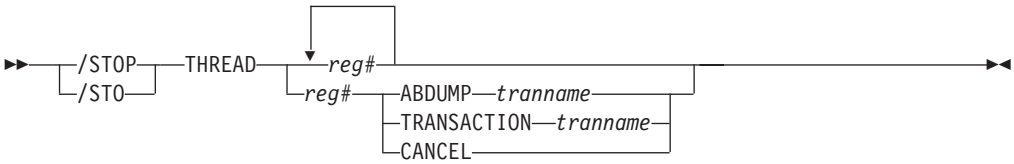
Table 269. Valid environments for the /STOP THREAD command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X

Table 269. Valid environments for the /STOP THREAD command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
ABDUMP	X	X	X
CANCEL	X	X	X
THREAD	X		X
TRANSACTION	X		X

Syntax



Keywords

For information about the keywords that are valid for the /STOP THREAD command, see the description about the same keywords under the /STOP REGION command.

Usage notes

The /STOP THREAD command is not valid for:

- Active CCTL threads
- BMPs

The THREAD ABDUMP command abnormally ends BMPs and DEDB utilities. If this command is used with CCTL threads, a U0474 abend results.

Examples

The following are examples of the /STOP THREAD command:

Example 1

Entry ET:

```
/DISPLAY A REGION
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS     CLASS
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
    2 CICS1A    DBT   IEFPROC   BMP255   ACTIVE
    3 CICS1A    DBT   IEFPROC   PLVAPZ12 ACTIVE
    1 CICS1A    DBT   IEPROC              AVAILABLE
      DBRCHTA1  DBRC
      DLICHTA1  DLS
*00082/142907*
```

Entry ET:

```
/STOP THREAD 2
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS0556I COMMAND REJECTED; DBCTL THREAD IS ACTIVE
```

Example 2

Entry ET:

```
/DISPLAY A REGION
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS     CLASS
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
    2 CICS1A    DBT   IEFPROC   BMP255   ACTIVE
    3 CICS1A    DBT   IEFPROC   PLVAPZ12 ACTIVE
    1 CICS1A    DBT   IEFPROC            AVAILABLE
      DBRCHTA1  DBRC
      DLICHTA1  DLS
*00082/143027*
```

Entry ET:

```
/STOP THREAD 1
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
```

Entry ET:

```
/DISPLAY A REGION
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS     CLASS
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
    2 CICS1A    DBT   IEFPROC   BMP255   ACTIVE
    3 CICS1A    DBT   IEFPROC   PLVAPZ12 ACTIVE
      DBRCHTA1  DBRC
      DLICHTA1  DLS
*00082/143055*
```

Example 3

Entry ET:

```
/DISPLAY A REGION
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS     CLASS
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
    2 CICS1A    DBT   IEFPROC   BMP255   ACTIVE
    3 CICS1A    DBT   IEFPROC   PLVAPZ12 ACTIVE
      DBRCHTA1  DBRC
      DLICHTA1  DLS
*00082/144731*
```

Entry ET:

```
/STOP THREAD 2 ABDUMP
```

Response ET:

```
/DFS058I STOP COMMAND IN PROGRESS
```

Response ET:

```
DFS554A CICS1A     00002 IEFPROC  BMP255   (3)        000,0474   20
/082 14:49:11    RTKN= CICS1   B3C81CB789F4BE83
```

Entry ET:

```
/DISPLAY A REGION
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS     CLASS
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
    3 CICS1A    DBT   IEFPROC   PLVAPZ12 ACTIVE
      DBRCHTA1  DBRC
      DLICHTA1  DLS
*00082/145038*
```

Related reference:

"/STOP REGION command" on page 533

/STOP TMEM command

Use the /STOP TMEM command to cause IMS to send an Open Transaction Manager Access (OTMA) command to OTMA clients to request that input be suspended for the specified transaction pipe name.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 550
- "Usage notes" on page 550
- "Examples" on page 550

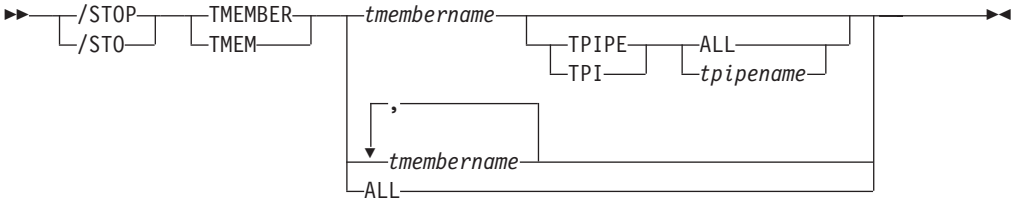
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 270. Valid environments for the /STOP TMEM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
TMEM	X		X
TPIPE	X		X

Syntax



Keywords

The following keyword is valid for the /STOP TMEM command:

TPIPE
When used with the /STOP command, TPIPE causes IMS to send an OTMA command to its OTMA clients to request that the input be suspended for the specified transaction pipe. IMS then stops sending output to the OTMA client. If the member specified is a super member, output is suspended for the super member's transaction pipe, but no OTMA command is sent. If the member specified is a regular member whose hold queue output is managed by a super member, IMS suspends output from the specified member's transaction pipe and it also suspends output from the super member's transaction pipe. An OTMA command is sent to the regular member's OTMA client. Output is only suspended on the IMS that processes the command. If output cannot be suspended for both the regular member's transaction pipe and the super member's transaction pipe, it is not suspended for either transaction pipe. The DFS058I COMMAND COMPLETED EXCEPT message is issued with the name of the regular member for which output could not be suspended

While processing the /STOP TMEMBER TPIPE command, IMS creates a temporary transaction pipe (if one does not already exist) with the stopped status. IMS sets the synchronization status for this transaction pipe when it sends or receives the first message for the transaction pipe.

While processing the /STOP TMEMBER TPIPE command, OTMA checks for wait status (WAIT_A, WAIT_H, and WAIT-SYNCPOINT) for the messages using the tpipe. If a wait status is found, OTMA clears the wait status by generating an internal NAK message. This NAK message for a send-then-commit (CM1) response will cause a U0119 pseudoabend for the transaction. However, the NAK message for a commit-then-send (CM0) response will wash back the response to the tpipe queue. The message in the tpipe queue can be retrieved again later.

After a /STOP TMEMBER *xxx* TPIPE ALL command is issued, newly created tpipes will not be stopped for either input or output.

The /STOP TMEMBER *xxx* TPIPE *xxx* command will not create a temporary tpipe if the tmember does not exist (DFS058I STOP COMMAND COMPLETED EXCEPT TPIPE *xxx* will be issued).

Usage notes

You can stop any number of individual tmembers or all tmembers. IMS then stops sending output to the OTMA client and prevents any further output from being sent to the client.

When the /STOP TMEM TPIPE command is issued, it will clear the wait states of all messages for the transaction pipe.

Examples

The following is an example of the /STOP TMEM command:

Entry ET:

```
/STO TMEMBER CLIENT1 TPIPE TPIPESY
```

Response ET:

```
DFS058I 15:38:03 STOP COMMAND COMPLETED   SYS3
DFS996I *IMS READY*  SYS3
```

/STOP TRAN command

Use the /STOP TRAN command to stop the queuing and scheduling of messages destined for a transaction or class of transactions, or to stop transaction scheduling by class. However, output can still be queued if it originates from the application program.

The /STOP TRAN command stops the scheduling of transactions; however, the transactions will continue to be processed until the limit count is reached. If the limit count is large, the processing interval will be long.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands" on page 552
- "Examples" on page 552

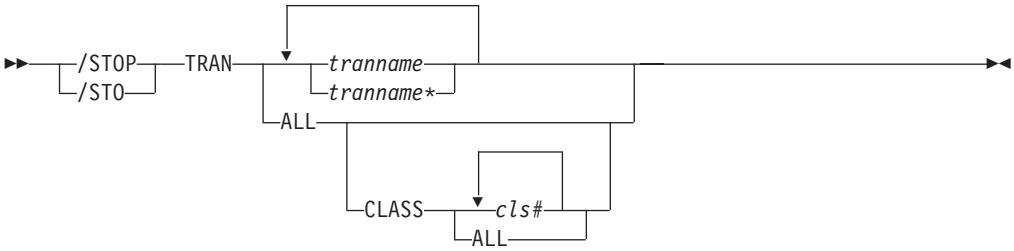
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 271. Valid environments for the /STOP TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
CLASS	X		X
TRAN	X		X

Syntax



Usage notes

If the region is already scheduled and waiting for the next message (wait-for-input mode) when the command is entered, a QC status (no more messages) is returned to the application (MPPs). If there are no more messages available for the region to process, the region is not allowed to wait for the next message. Instead, a QC status is returned to the application (MPPs).

A batch message processing region (BMP) scheduled against wait-for-input (WFI) transactions returns a QC status code (no more messages) for /PSTOP REGION, /DBD, /DBR, or /STA commands only.

In a shared-queues environment, if you issue a /STOP TRAN command for a transaction that is not defined on that IMS subsystem, IMS creates an SMB if the Output Creation user exit routine indicates the destination is a valid transaction. The SMB is marked as “dynamic”.

A dynamic SMB created by a /STOP TRAN command can only be used to queue messages for the transaction and place the messages on the shared queues. The transaction cannot be scheduled or assigned. IMS does process checkpoints for the transaction, but does not save them across an IMS restart if they do not have a valid status.

In a shared-queues environment, the /STOP TRAN command will result in IMS deregistering interest for the transaction, which indicates that the transaction cannot be scheduled at that IMS.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP TRAN command and the IMS type-2 commands that perform similar functions.

Table 272. Type-2 equivalents for the /STOP TRAN command

Task	/STOP TRAN command	Similar IMS type-2 command
Stops the queuing and scheduling of messages destined for a transaction.	/STOP TRAN *tranname*	UPDATE TRAN NAME(*tranname*) STOP(Q,SCHD)

Examples

The following are examples of the /STOP TRAN command:

Example 1 for /STOP TRAN command

Entry ET:

```
/STOP TRANSACTION ALL CLASS 6
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: All transactions associated with class 6 will be marked as stopped and all class 6 transactions are no longer available for scheduling. All message processing regions currently processing class 6 transactions are allowed to run until the processing limit count is reached or the input queue contains no more messages.

Example 2 for /STOP TRAN command

Entry ET:

```
/STOP TRANSACTION PIT, SEED
```

Response ET:

```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: Transaction codes PIT and SEED are stopped.

Related reference:

"UPDATE TRAN command" on page 795

/STOP USER command

The /STOP USER command requires the ISC user to stop or the signed on user to stop and sign off. The USER parameter can be generic and applies only to users that already exist.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 554

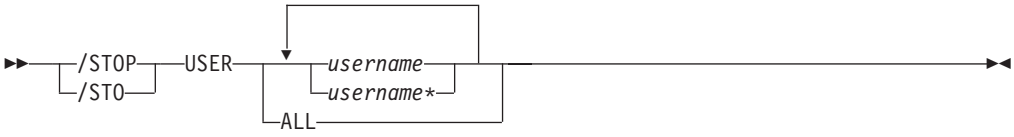
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 273. Valid environments for the /STOP USER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
USER	X		X

Syntax



Usage notes

For ISC users, the /STOP USER command specifies the ISC user that is to be made unavailable for allocation until a /START USER command is issued.

For signed on users, the /STOP USER command should specify the user structure name to prevent future signons until a /START USER command is issued.

The /STOP USER command is supported for users that do not yet exist. It causes the user to be created and stopped, which prevents the dynamic user from signing on.

The /STOP USER command will not cause the user to be signed off if the associated node is not active or the associated node is not in session.

The /STOP USER command for an ETO user session resets status that is not significant such as preset mode, test mode, lock lterm, pstop lterm, and purge lterm. For Fast Path input response mode, you must also issue the /START USER command before the mode is reset.

The /STOP USER command for ETO users also takes other actions depending on the recovery settings for the user:

RCVYCONV=NO
: /STOP USER causes any IMS conversations (active and held) for an ETO user to be terminated. Any conversational message that is queued or being processed will have its output response message delivered asynchronously.

RCVYFP=NO
: /STOP USER causes Fast Path status and messages for an ETO user to be discarded.

RCVYRESP=NO
: /STOP USER resets full-function response mode.

If global resource information is kept in Resource Manager, the /STOP USER command sets a global stop signon status for the user and prevents the user from signing on anywhere in the IMSplex. If global resource information is not kept in Resource Manager, /STOP USER creates the user, if it does not exist in an ETO environment, and sets stop status for the local user. If the user does not exist in a non-ETO environment, the /STOP USER command is rejected.

Examples

The following is an example of the /STOP USER command:

Entry ET:

```
/DISPLAY USER IMS*
```

Response ET:

```
USER     ENQCT DEQCT   QCT   SYS3
IMSUS06      0     0     0 ALLOC(DTSLU602)
IMSUS04      0     0     0 ALLOC(DTSLU603)
IMSUS03      0     0     0 ALLOC(DTSLU601)
IMSUS02      0     0     0 ALLOC(DTSLU202)
IMSUS01      0     0     0 ALLOC(DTSLU201)
IMSUS09    N/A   N/A   N/A ALLOC(ENDS02  ) STATIC
IMSUS08    N/A   N/A   N/A ALLOC(ENDS01  ) STATIC
IMSUS11    N/A   N/A   N/A ALLOC(ENDS03  ) STATIC
IMSUS10    N/A   N/A   N/A ALLOC(OMSSLU2A) STATIC
*91091/111454*
```

Entry ET:

```
/STOP USER IMSUS01 IMSUS02
```

Response ET:

```
DFS058I 11:16:24 STOP COMMAND COMPLETED
```

Entry ET:

```
/STOP USER HELLO%
```

Response ET:

```
DFS3633 11:18:25 GENERIC PARAMETER RESOURCES NOT FOUND, NO ACTION TAKEN
```

Entry ET:

```
/DISPLAY USER IMSUS01 IMSUS02
```

Response ET:

```
USER     ENQCT DEQCT   QCT
IMSUS01      0     0     0 STOPPED
IMSUS02      0     0     0 STOPPED
*91091/111727*
```

/STOP VGR command

Use the /STOP VGR command to cause the IMS subsystem to drop out of a generic resources group. This command is rejected if the VTAM ACB is closed (usually the result of a /STOP DC command).

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 274. Valid environments for the /STOP VGR command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
VGR	X		X

Syntax

Usage notes

While this command prevents VTAM from routing new sessions using a generic resource name to the IMS subsystem, it does not affect existing sessions, and affinities remain (until terminated through normal processing).

/STOP WADS command

Use the /STOP WADS command to indicate that a WADS is to be removed from the pool of available WADS. IMS does not enable the active WADS (if WADS mode is single), or the active WADS pair (if WADS mode is dual), to be stopped. wads# must be 0 through 9.

Subsections:

- "Environment" on page 556
- "Syntax" on page 556

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 275. Valid environments for the /STOP WADS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
WADS	X	X	X

Syntax

/STOP XRCTRACK command

The /STOP XRCTRACK command results in calls to the log router to initiate or terminate XRC tracking. It is only valid on a tracking IMS system.

Subsections:

- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 276. Valid environments for the /STOP XRCTRACK command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DC	X	X	X

Syntax

Chapter 24. /SWITCH command

Use the /SWITCH command to switch active data sets or to change between the active and alternate systems. Certain combinations of keywords are valid only in the active or alternate systems, as shown in the format in the syntax diagram.

This command can be issued to an IMSplex using the Batch SPOC utility.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 558
- "Examples" on page 559

Environment

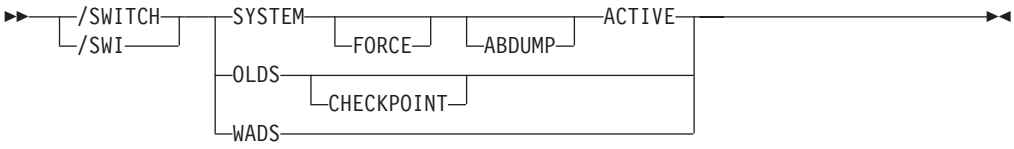
The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 277. Valid environments for the /SWITCH command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/SWITCH	X	X	X
ABDUMP	X		X
ACTIVE	X		X
BACKUP	X		X
CHECKPOINT	X	X	X
FORCE	X		X
OLDS	X	X	X
SYSTEM	X		X
WADS	X	X	X

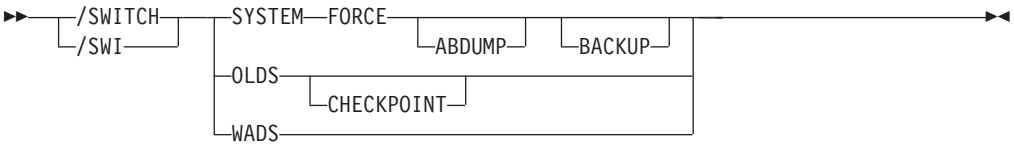
Syntax

/SWITCH for an active XRF subsystem



/SWITCH for an alternate XRF subsystem

© Copyright IBM Corp. 1974, 2015



Keywords

The following keywords are valid for the /SWITCH command:

SYSTEM
Requests a takeover by the alternate system from the current active system in an XRF environment.

/SWITCH SYSTEM without the FORCE keyword causes the active system to attempt to gracefully quiesce before the alternate system becomes active. System activity currently in progress is allowed to complete. New activity is queued. This disables surveillance on the active system, but not the alternate. Therefore, surveillance (if enabled) will eventually trigger a takeover if it does not eventually quiesce.

Unless the optional FORCE keyword is specified, the command is only operable when entered on the active system.

FORCE
Causes an immediate termination of the active system, forcing the alternate system to become the active.

ABDUMP
Results in a diagnostic dump of the active system when entered from either the active system or the alternate system (if it is on the same processor as the active system).

ACTIVE, BACKUP
Indicates the system on which the command is being entered. The keyword ACTIVE is required when the command is entered on an active system. The keyword BACKUP is optional when the command is entered on an alternate system. /SWITCH SYSTEM FORCE, without the ACTIVE keyword, can only be entered on an alternate system. This prevents the inadvertent abend of a newly created active system that is mistakenly assumed to still be the alternate system.

OLDS, CHECKPOINT
Causes switching of the active log data set. This log switch capability is identical to that provided with /DBDUMP and /DBRECOVERY commands. You can specify the CHECKPOINT keyword to take a simple checkpoint after the active log data set has been switched to the next OLDS. The /SWITCH OLDS CHECKPOINT command operates in all IMS environments.

WADS
Causes switching of the active write-ahead log data set. If you are using dual logging for the WADS, this command causes IMS to use the next available WADS pair.

This command is rejected if no unused WADS is available, or for dual logging, if no unused pair of WADS is available.

Examples

The following are examples of the /SWITCH command:

Example 1 for /SWITCH command

Entry ET:

```
/SWITCH OLDS
```

Response ET:

```
DFS3257I ONLINE LOG NOW SWITCHED
DFS058I 17:10:51 SWITCH COMMAND COMPLETED
```

Entry ET:

```
/SWITCH OLDS CHECKPOINT
```

Response ET:

```
DFS3257I ONLINE LOG NOW SWITCHED - FROM DFSOLP01 TO DFSOLP02
DFS058I 17:12:53 SWITCH COMMAND COMPLETED

DFS2719I MSDB CHECKPOINT WRITTEN TO MSDBCP2
DFS994I *CHKPT 91057/171254**SIMPLE*
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:    1
DFS3804I LATEST RESTART CHKPT: 91057/132000, LATEST BUILDQ CHKPT:
91057/132414
```

Example 2 for /SWITCH command

Entry ET (Master Terminal for active system IMSA):

```
/SWITCH SYSTEM FORCE
```

Response ET:

A response message is not returned for the /SWITCH SYSTEM FORCE command. Any further input to the master terminal of the active system is inhibited.

Response RT (z/OS console for active system IMSA):

The z/OS console for the active system will show a user 0604 abend in progress for IMSA. If the ABDUMP keyword had been included on the /SWITCH command, the 0604 abend would be accompanied by a diagnostic dump of the active system.

Response RT (master terminal for alternate system IMSB):

The following figure is a screen that shows some of the messages associated with the beginning of takeover on the alternate system.

```
02/05/15 15:28:27  RSENAME: DFSRSENM  BACKUP   TAKEOVER IN PROGRESS  IMSB
DFS3890I 15:27:18 TAKEOVER REQUESTED
DFS970I 15:28:05 UNEXPECTED  STATUS   ,NODE APPLA  ,USER  N/A  ,SEND  ,RC
=14,FDB2=13,NSECIT =29,SENSE=00000000,REASON=00
DFS3257I ONLINE LOG CLOSED ON DFSOLP00
DFS3891I 15:28:18 TAKEOVER IN PROGRESS

 DFS2591I NO MSDB HEADERS FOUND, IMAGE COPY LOAD IGNORED
DFS3839I 14:26:46 XRF INITIAL DB PRE-OPEN COMPLETE.
DFS3838I 14:28:41 XRF INITIAL DC PRE-OPEN COMPLETE.
--------------------------------------------------------------------------------









                                                      PASSWORD:
```

Figure 1. Alternate system at start of takeover

Intermediate screens are not shown. They would indicate such takeover functions as:

- Enabling of dependent region processing
- IRLM takeover
- Backout processing
- Draining of suspend queue
- Session switching

The following figure is a screen that show takeover is complete.

```
02/05/15 15:30:59  RSENAME: DFSRSENM  ACTIVE  AWAITING I/O PREVENTION  IMSB
 DFS994I *CHKPT 85135/152931**SIMPLE**
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:   11
DFS3804I LAST CHKPT ID VALID FOR RESTART: 85135/152931-BUILDQ: 85135/142629


DFS994I TAKEOVER COMPLETED.
DFS3859I 15:29:19 PRIORITY 4 SESSIONS SWITCHED.
DFS3860I 15:29:19 ALL TERMINAL SESSIONS SWITCHED.
--------------------------------------------------------------------------------









                                                      PASSWORD:
```

Figure 2. Newly created active system after takeover

Takeover is complete and the alternate system is now an active system. The XRF environment status line indicates that the newly created active system is running in I/O toleration mode (awaiting I/O prevention).

Related concepts:

Takeover phase of the XRF process (System Administration)

Chapter 25. TERMINATE commands

Use the TERMINATE commands to terminate a global online change or to stop one or more HALDB OLRs that are in progress.

- "TERMINATE OLC command"
- "TERMINATE OLREORG command" on page 575

TERMINATE OLC command

When the TERMINATE OLC (stop online change) command is issued by an IMS command master that is running with RM services (RMENV=Y), the command terminates a global online change and coordinates with all of the IMS systems in the IMSplex.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "TERMINATE OLC error handling" on page 564
- "Output fields" on page 566
- "Return, reason, and completion codes" on page 567
- "Examples" on page 574

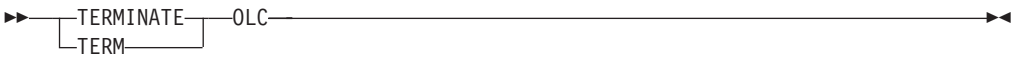
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the TERMINATE OLC command can be issued.

Table 278. Valid environments for the TERMINATE OLC command

Command / Keyword	DB/DC	DBCTL	DCCTL
TERMINATE OLC	x	x	x

Syntax



Usage notes

If an IMS is not running with RM services (RMENV=N), the TERMINATE OLC command terminates an online change for that IMS only. If an IMS system is not running with RM, each IMS must have a unique OLCSTAT data set which cannot be shared. If the OLCSTAT data set contains the name of an IMS other than the one that is processing the online change, TERMINATE OLC is rejected because the OLCSTAT data set is invalid for the environment. To determine which IMS member names are invalid, issue the QUERY OLC command to display the contents of the OLCSTAT data set. You can use the OLC utility, DFSUOLC0, to correct the data set.

© Copyright IBM Corp. 1974, 2015

Each IMS system that does not have RM services and participates in global online change, must separately issue the TERMINATE OLC command. In a non-RM environment, if more than one IMS is specified in the route list for the TERMINATE OLC command or the default of route all is specified, online change is only performed for the IMS command master. To determine which IMS systems are defined with RMENVNO, issue a QUERY MEMBER SHOW(ATTRIB) command.

This command can be specified only through the OM API. OM sends the TERMINATE OLC command to an IMS in the IMSplex.

This command can be issued to an IMSplex using the Batch SPOC utility.

The TERMINATE OLC command can be used to abort an IMSplex-wide global online change initiated by a INITIATE OLC PHASE(PREPARE) command, before the online change is successfully committed with a INITIATE OLC PHASE(COMMIT) command.

The TERMINATE OLC command can be used to abort an online change after an INITIATE OLC PHASE(COMMIT) failure that occurs before the OLCSTAT data set is updated. Once the commit process has updated the OLCSTAT data set, the online change is considered to be successful and cannot be aborted.

A TERMINATE OLC command that aborts a global online change is similar to the /MODIFY ABORT command, except that it applies to all of the IMS systems in an IMSplex that are participating in the global online change. The TERMINATE OLC command can be used to terminate an INIT OLC TYPE(ACBMBR) command in progress. If the updated members have not been committed, the TERM OLC command will delete the new versions of the updated members in the active ACBLIB. The TERM OLC command for a TYPE(ACBLIB) will log a X'7010' log record indicating that the member OLC has been terminated.

The TERMINATE OLC command is not supported if local online change is enabled. The TERMINATE OLC command is rejected if the IMS to which the command is routed does not support global online change. If this occurs and there is an IMS that supports global online change, the user must route the command to a specific IMS that supports global online change.

The TERMINATE OLC command is invalid on the XRF alternate, RSR tracker, and FDBR system.

The OM command timeout default of 300 seconds (5 minutes) might not be enough time for the online change phase to complete. You might need to specify a timeout value on the command based on the needs of the installation. To change the timeout value, use the TIMEOUT parameter of the CSLOMCMD command request or specify a Wait (timeout) value when you issue the TERMINATE OLC command from TSO SPOC.

The command syntax for this command is defined in XML and is available to automation programs which communicate with OM.

TERMINATE OLC error handling

Errors unique to the processing of this command are returned as a completion code. A completion code is returned for each action against an individual resource.

The TERMINATE OLC command might result in an error that leaves IMS systems in various online change states. Correct the error. Issue the QUERY MEMBER TYPE(IMS) SHOW(STATUS) command to display the online change state of all the IMS systems in the IMSplex. Evaluate the QUERY MEMBER TYPE(IMS) output to help you determine what to do:

- None of the IMS systems in an online change state

  The TERMINATE OLC command succeeded or was not applicable. No further action needs to be taken.
- Some of the IMS systems in a prepare complete state

  The online change is not committed. Correct the problem that caused TERMINATE OLC to fail, then try the TERMINATE OLC command again.

  The IMS systems that are in an online change state remain in an online change state until you abort the online change.
- All IMS systems in a prepare complete state

  The online change is not committed. Correct the problem that caused TERMINATE OLC to fail, then try the TERMINATE OLC command again.

  The IMS systems that are in an online change state remain in an online change state until you abort the online change.
- Some IMS systems in prepare complete and commit phase 1 complete state

  The commit phase failed before the master updated the OLCSTAT data set, so the online change is not committed. Correct the problem that caused TERMINATE OLC to fail and try the TERMINATE OLC command again.

  The IMS systems that are in an online change state remain in an online change state until you abort the online change.
- All IMS systems in commit phase 1 complete state

  If the commit phase failed before the master updated the OLCSTAT data set, the online change is not committed. Correct the problem that caused TERMINATE OLC to fail and try the TERMINATE OLC command again.

  If the commit phase failed after the master updated the OLCSTAT data set, the online change is committed. The TERMINATE OLC command is not permitted. You must correct the problem that caused the commit command to fail and try the INITIATE OLC PHASE(COMMIT) command again.

  The IMS systems that are in an online change state remain in an online change state until you abort the online change or commit the online change.

  You can determine whether the OLCSTAT data set has been updated by the modify ID. Issue the QUERY OLC LIBRARY(OLCSTAT) SHOW(MODID) command. Check if the modify ID returned is different from the modify ID returned by the INITIATE OLC PHASE(PREPARE) command, or the modify ID returned by a QUERY OLC LIBRARY(OLCSTAT) SHOW(MODID) command issued before the INITIATE OLC PHASE(COMMIT) command.
- Some IMS systems in commit phase 1 complete state and some in commit phase 2 complete state

  The online change is committed. The TERMINATE OLC command is not permitted. You must correct the problem that caused the commit command to fail and try INITIATE OLC PHASE(COMMIT) again.

  The IMS systems that are in an online change state remain in an online change state until you finish the online change with an INITIATE OLC PHASE(COMMIT) command.
- All IMS systems in commit phase 2 complete state

The online change is committed. The TERMINATE OLC command is not permitted. You must correct the problem that caused the commit command to fail and try INITIATE OLC PHASE(COMMIT) again.

The IMS systems that are in an online change state remain in an online change state until you finish the online change with an INITIATE OLC PHASE(COMMIT) command.

- Some IMS systems in commit phase 2 complete state and some not in online change state

  The online change is committed. The TERMINATE OLC command is not permitted. You must correct the problem that caused the commit command to fail and try INITIATE OLC PHASE(COMMIT) again.

  The IMS systems that are in an online change state remain in an online change state until you finish the online change with an INITIATE OLC PHASE(COMMIT) command.

Errors unique to the processing of this command are returned as a completion code. A completion code is returned for an IMS participating in the online change phase.

Output fields

The following table shows the output fields for a TERMINATE OLC command. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keywords on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 279. Output fields for the TERMINATE command

Short label	Keyword	Meaning
MBR	N/A	IMSplex member that built the line of output. IMS identifier of the IMS that was master of the abort phase. IMS identifier is always returned.
IMSMBR	N/A	IMS member that performed the global online change phase. The IMS member name is always returned.
CC	N/A	Completion code from the IMS member that performed the online change phase. Completion code is always returned.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
ERRT	N/A	Error text associated with a nonzero completion code returned by the IMS member that performed the online change phase. Error text might be returned if the completion code is nonzero.

Return, reason, and completion codes

The OM return and reason codes that might be returned as a result of this command are standard for all commands entered through the OM API. Refer to the OM CSLOMCMD Return and Reason code section for the list of codes and their meanings.

An IMS return and reason code is returned to OM by the TERMINATE OLC command.

Some reason codes are accompanied by a complete list of IMS systems and completion codes. The reason code meaning indicates whether a list is returned. A partial list of IMS systems and completion codes might be returned with any TERMINATE OLC error reason code, if any output was built before the error was detected.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 280. Return and reason codes for the TERMINATE OLC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The TERMINATE OLC command completed successfully. The TERMINATE OLC command is applied to all of the IMS systems listed in the OLCSTAT data set. All of the IMS systems in the IMSplex are no longer in an online change state. An output line is built for each IMS listed in the OLCSTAT data set. Each output line contains the IMS member name and a completion code of zero.
X'00000004'	X'0000100C'	The TERMINATE OLC command completed successfully, but was not applicable to one or more IMS systems for acceptable reasons. The TERMINATE OLC command applies to all of the IMS systems listed in the OLCSTAT data set. An output line is built for each IMS listed in the OLCSTAT data set. Each output line contains the IMS member name and a completion code. A nonzero completion code may be accompanied by error text. One or more of the IMS systems contain a completion code that indicates that the terminate online change did not apply to this IMS, such as the IMS state is abended, the IMS state is shutdown, or this IMS is already in the correct online change state. The TERMINATE OLC completion code table contains the list of completion codes and error text that can be returned by the TERMINATE OLC command.

Table 280. Return and reason codes for the TERMINATE OLC command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003000'	The TERMINATE OLC command is successful for at least one IMS but not all IMS systems. The TERMINATE OLC command applies to all IMS systems listed in the OLCSTAT data set. An output line is built for each IMS listed in the OLCSTAT data set. Each output line contains the IMS member name and a completion code. A nonzero completion code might be accompanied by error text. One or more of the IMS systems returned an error completion code. The TERMINATE OLC completion code table contains the list of completion codes and error text that can be returned by the TERMINATE OLC command. If the TERMINATE OLC command fails for one or more IMS systems, correct the problem and issue the TERMINATE OLC command again. For more details, see "TERMINATE OLC error handling" on page 564.
X'0000000C'	X'00003004'	The TERMINATE OLC command failed for all of the IMS systems. The TERMINATE OLC command applies to all of the IMS systems listed in the OLCSTAT data set. An output line is built for each IMS listed in the OLCSTAT data set. Each output line contains the IMS member name and a completion code. A nonzero completion code might be accompanied by error text. All of the IMS systems returned an error completion code. The TERMINATE OLC completion code table contains the list of completion codes and error text that can be returned by the TERMINATE OLC command. If the TERMINATE OLC command fails for one or more IMS systems, correct the problem and issue the TERMINATE OLC command again. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000010'	X'00004004'	The TERMINATE OLC command failed because there is no CQS. RM attempted to access the process resource on the resource structure, but it failed because CQS is not available. The online change phase might have succeeded on one or more IMS systems. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000010'	X'0000400C'	The TERMINATE OLC command failed because it is invalid for an XRF alternate.
X'00000010'	X'00004014'	The TERMINATE OLC command failed because it is invalid for an RSR tracker.
X'00000010'	X'00004018'	The TERMINATE OLC command failed because the RM resource structure is not available. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000010'	X'0000401C'	The TERMINATE OLC command failed because it is invalid for an FDBR region.
X'00000010'	X'00004100'	The TERMINATE OLC command is rejected because the resource structure is full. RM failed trying to create the process resource on the resource structure. One or more IMS systems might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 564.

Table 280. Return and reason codes for the TERMINATE OLC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004104'	The TERMINATE OLC command failed because RM is not available. The online change phase might have succeeded on one or more IMS systems. Either there is no RM address space, or RM is active but not registered to SCI because CQS or the resource structure is not available. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000010'	X'00004108'	The TERMINATE OLC command failed because SCI is not available. One or more IMS systems might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000010'	X'0000410C'	The TERMINATE OLC command is rejected, because global online change is not enabled. Local online change is enabled. Use the /MODIFY command for local online change. If your IMSplex is made up of some IMS systems that support global online change and some that support local online change, route the TERMINATE OLC command to an IMS that is enabled for global online change.
X'00000010'	X'00004110'	The TERMINATE OLC command is rejected, because the command does not apply to the online change state of the command master. TERMINATE OLC is rejected if the command master is not in an online change state. TERMINATE OLC is rejected if the command master has already committed the online change. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000010'	X'0000412C'	The OLCSTAT data set contains the name of an IMS other than the IMS processing the online change. Use DFSUOLC0 to correct the data set.
X'00000010'	X'00004114'	The TERMINATE OLC command failed because of an error accessing the OLCSTAT data set. One or more IMS systems in the IMSplex might still be in an online change state. A DFS2843 message is sent to the OM output exit as unsolicited output. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000010'	X'00004118'	The TERMINATE OLC command failed because of an error allocating the OLCSTAT data set. One or more IMS systems in the IMSplex might still be in an online change state. A DFS2848 message is sent to the OM output exit as unsolicited output. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000010'	X'0000411C'	The TERMINATE OLC command failed because of an error in the OLCSTAT data set contents. One or more of the values is invalid. A DFS2844 message is sent to the OM output exit as unsolicited output.
X'00000010'	X'00004120'	The TERMINATE OLC command is rejected because an online change phase is already in progress on this IMS, which might be INITIATE OLC, TERMINATE OLC, or /DISPLAY MODIFY.
X'00000014'	X'00005000'	The TERMINATE OLC command is rejected because an `IMODULE GETSTOR` storage request failed.

Table 280. Return and reason codes for the TERMINATE OLC command (continued)

Return code	Reason code	Meaning
X'00000014'	X'00005004'	The TERMINATE OLC command failed because a DFSOCMD response buffer could not be obtained. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000014'	X'00005100'	The TERMINATE OLC command failed because of an RM error. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000014'	X'00005104'	The TERMINATE OLC command failed because of a CQS error. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000014'	X'00005108'	The TERMINATE OLC command failed because of an SCI error. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 564.
X'00000014'	X'00005FFF'	The TERMINATE OLC command failed because of an internal IMS error. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 564.

The following table contains the completion codes that can be returned on a TERMINATE OLC command, the meaning of the completion code, and any error text associated with the code.

Table 281. Completion codes for the TERMINATE OLC command

Completion code	Meaning	ERROR TEXT (uppercase)
0	The online change commit or abort phase completed successfully.	
1	The online change type does not apply to this IMS. For example, an ACBLIB online change does not apply to a DCCTL IMS. This IMS does nothing.	
2	The online change phase was not attempted by this IMS for one of the following reasons: • The online change phase master encountered an error and did not direct this IMS to perform the online change phase.	

Table 281. Completion codes for the TERMINATE OLC command (continued)

Completion code	Meaning	ERROR TEXT (uppercase)
3	The online change for this IMS is already completed or terminated. This IMS coordinates the termination of the global online change, but this IMS does not have to do anything locally. The TERMINATE OLC command completed successfully and cleaned up information about the global online change, if there was any. An example of this is the information that RM keeps to manage the global online change.	
58	An IMS is not registered to RM. An OLCSTAT data set contains an IMS that is not registered to RM. Terminate fails for that IMS.	
60	IMODULE GETMAIN storage error.	
61	BCB storage error.	
62	HIOP storage error.	
63	WKAP storage error.	
80	Data set error.	Function (8 char), ddname (8 char), return code (8 bytes), and error detail (8 char). Function can be one of the following: • OPEN Data set open error. • READ Data set read error. DDname can be OLCSTAT. Return code is the data set service return code. Reason code is the data set service reason code.
90	Internal error.	Module name that detected internal error (8 char), unused (8 char), return code or function code (8 bytes), and error detail (8 char).

Table 281. Completion codes for the TERMINATE OLC command (continued)

Completion code	Meaning	ERROR TEXT (uppercase)
91	The online change commit phase 2 or abort phase timed out before this IMS responded to the online change commit phase 2 or abort phase. The commit phase 2 or abort might have succeeded on this IMS. Issue QUERY MEMBER TYPE(IMS) to determine the online change state of this IMS.	

Table 281. Completion codes for the TERMINATE OLC command (continued)

Completion code	Meaning	ERROR TEXT (uppercase)
B2	IMS state error.	IMS state error (32 char). The IMS state can be one of the following: • ABENDED This IMS ended abnormally since the last successful online change. Online change is terminated on this IMS. • NOT-REACHABLE The online change phase is rejected because this IMS is NOT-REACHABLE. The SCI on the OS image where this IMS is active is down. Restart the SCI and issue the INITIATE OLC or TERMINATE OLC command again. • OLC ALREADY COMMITTED The online change terminate is rejected because online change is already committed. All IMS systems have completed commit phase 1 and the OLCSTAT data set was updated. • OLC NOT IN PROGRESS The IMS is not in an online change state. The request to terminate the online change does not apply to this IMS. • OLC PHASE IN PROGRESS The online change phase is rejected because this IMS has an online change phase already in progress. • RESTART NOT COMPLETE This IMS initialized before the online change was initiated, but has not completed restart. The online prepare or abort phase is rejected as long as this IMS is in this state. Cancel this IMS, then abort the online change before attempting the online change prepare phase again. • SHUTDOWN This IMS shut down normally since the last successful online change. Online change is terminated on this IMS.
14A	Command failed for the IMS because its SCI was not active. When SCI is not active, the IMS is not reachable.	NOT-REACHABLE
14B	Command failed for the IMS because it shut down normally during command processing.	SHUTDOWN

Table 281. Completion codes for the TERMINATE OLC command (continued)

Completion code	Meaning	ERROR TEXT (uppercase)
14C	Command failed for the IMS because it terminated abnormally during command processing.	ABENDED

Examples

TSO SPOC input:

```
TERMINATE OLC
```

TSO SPOC output:

```
MbrName  Member     CC
IMS3     IMS2        0
IMS3     IMS3        0
IMS3     SYS3        0
```

OM API input:

```
CMD (TERMINATE OLC)
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 15:49:27.197919</statime>
<stotime>2002.16315:49:27.712209</stotime>
<staseq>B7C4ADFC0D4DF841</staseq>
<stoseq>B7C4ADFC8ADD1F45</stoseq>
<rqsttkn1>USRT011 10084927</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS2    ">
<typ>IMS     </typ>
<styp>DBDC   </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
</mbr>
<mbr name="SYS3    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
</mbr>
</cmderr>
<cmd>
<master>IMS3    </master>
<userid>USRT011 </userid>
<verb>TERM</verb>
<kwd>OLC  </kwd>
<input>TERMINATE OLC</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"    llbl="MbrName" scope="LCL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="left" />
<hdr slbl="IMSMBR" llbl="Member"  scope="LCL" sort="a" key="1" scroll="no"  len="8"
```

```
dtype="CHAR" align="left" />
<hdr slbl="CC"     llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT"  align="right" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS3    ) IMSMBR(SYS3    ) CC(   0) </rsp>
<rsp>MBR(IMS3    ) IMSMBR(IMS2    ) CC(   0) </rsp>
<rsp>MBR(IMS3    ) IMSMBR(IMS3    ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Global online change was aborted for the IMSplex after a successful INITIATE OLC PHASE(PREPARE) command. Global online change was successfully terminated.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

TERMINATE OLREORG command

The TERMINATE OLREORG command is used to stop one or more HALDB online reorganizations (OLRs) that are in progress.

The command stops OLR processing at a unit of reorganization boundary, unless the OPTION keyword is specified.

This command supports the type-1 command format and the type-2 command format. The type-1 command format is /TERMINATE OLREORG. The type-1 command response is returned as a DFS0725I pre-edit message. The type-2 command format is TERMINATE OLREORG. The type-2 command response is returned as XML and is available to automation programs.

Subsections:

- "Environment"
- "Syntax" on page 576
- "Keywords" on page 576
- "Usage notes" on page 577
- "Output fields" on page 577
- "Return, reason, and completion codes" on page 578
- "Examples" on page 578

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the TERMINATE OLREORG command can be issued.

Table 282. Valid environments for the TERMINATE OLREORG command, keywords, and parameters

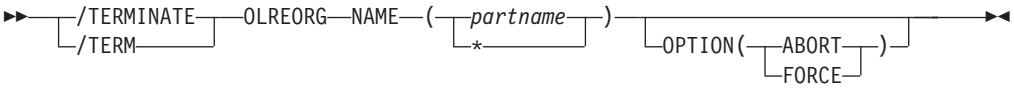
Command / Keywords	DB/DC	DBCTL	DCCTL
TERMINATE OLREORG	X	X	
NAME	X	X	

Table 282. Valid environments for the TERMINATE OLREORG command, keywords, and parameters (continued)

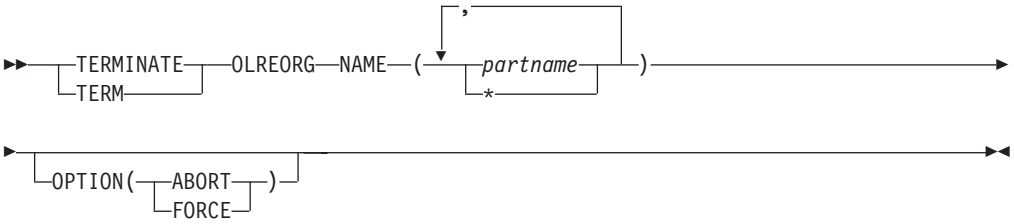
Command / Keywords	DB/DC	DBCTL	DCCTL
OPTION	X	X	

Syntax

Type-1 command syntax



Type-2 command syntax



Keywords

The following keywords are valid for the TERMINATE OLREORG command:

NAME()
Specifies the names or name of a HALDB partition for which the OLR is to be stopped. You can specify only PHDAM or PHIDAM HALDB partition names. A parameter with the wildcard character (*) is not allowed, except as NAME(*) for all defined HALDB partitions.

For the type-2 version of this command, you can specify one or more HALDB partition names.

For the type-1 version of the command, you can specify only one partition name.

OPTION()
Allows you to specify the FORCE or ABORT options.

ABORT
Causes the HALDB OLR to be stopped immediately, possibly with a completion code of abend U0474, without waiting for the current unit-of-reorganization to complete. Backout may be required depending on the state of the online reorganization at termination.

FORCE
Specifies that the HALDB OLR for the named *part name* is to be stopped when the next record boundary is encountered. All of the moved data up to that point is committed to DASD and no backout is required.

Attention: If the HALDB OLR is stopped prior to completion, the OPTION(NODEL) is not retained and must be specified on the INITIATE OLREORG command that is issued to resume the stopped online reorganization or on the UPDATE OLREORG command.

Usage notes

If /TERMINATE OLREORG or TERMINATE OLREORG is issued from OM API, it is treated as a type-2 command. Therefore, if you issue TERM OLREORG as a type-2 command from an OM API, the only valid command verb form is TERM or TERMINATE. Similarly, if /TERMINATE OLREORG is issued from a terminal, it is treated as a type-1 command. Therefore, you can issue /TERMINATE OLREORG using the first three command characters, such as /TER OLREORG.

This command can be issued to an IMSplex using the Batch SPOC utility.

The TERMINATE OLREORG command causes HALDB OLR to be stopped for the specified HALDB partitions. After a HALDB OLR is terminated, it is no longer active and it does not have an owning IMS. The partitions remain in cursor-active status until the online reorganization is resumed with an INITIATE OLREORG command and completes, or until you run an offline reorganization.

Command responses for /TERMINATE OLREORG

When you issue the /TERMINATE OLREORG command as a type-1 command, the command response is returned in a message format.

When the command completes successfully, the message, DFS0725I, is returned to the system console and to the master terminal with a completion code of 0. If the command results in an error, a non-zero completion code or an error message is returned to the master terminal and system console.

```
DFS0725I INITIATE|UPDATE|TERMINATE OLREORG COMMAND FOR DB dbnamexx COMPLETE.
CC=  nn
  where: dbnamexx is the HALDB partition name entered on the command
         nn       is the completion code
```

Output fields

This section describes the responses from the OM API for the TERMINATE OLREORG command. The following table shows the TERMINATE OLREORG output fields. The columns in the table are as follows:

Short label
Contains the short label that is generated in the XML output. This field does not apply to the /TERMINATE command.

Show Keyword
Identifies the command keyword that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 283. Output fields for TERMINATE OLREORG command

Short label	Show Keyword	Meaning
PART	N/A	Partition name.
MBR	N/A	The IMS that built the command response line.

Table 283. Output fields for TERMINATE OLREORG command (continued)

Short label	Show Keyword	Meaning
CC	N/A	Completion code.

Return, reason, and completion codes

The OM return and reason codes that might be returned as a result of the TERMINATE OLREORG command are standard for all commands that are entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 284. Return and reason codes for the TERMINATE OLREORG command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The TERMINATE OLREORG command completed successfully.
X'00000004'	X'00001010'	No matches found for filter.
X'0000000C'	X'00003000'	At least one request was successful.
X'0000000C'	X'00003004'	None of the requests was successful.
X'00000010'	X'00004014'	Command issued on an RSR tracker.
X'00000010'	X'0000400C'	Command issued on an XRF alternate.
X'00000014'	X'00005000'	A GETMAIN error occurred.

The following table includes an explanation of the completion codes. Errors unique to the processing of TERMINATE OLREORG command are returned as completion codes. A completion code is returned for each action against a HALDB partition.

Table 285. Completion codes for the TERMINATE OLREORG command

Completion code	Meaning
0	The TERMINATE OLREORG command completed successfully for the partition.
10	Resource name is invalid.
14	Resource is not a partition name.
1C	Resource is a partitioned secondary index.
24	No HALDB OLR is in progress.

Examples

The following are examples of the TERMINATE OLREORG command:

Example 1 for /TERMINATE OLREORG command

Entry ET:

```
/TERM OLREORG NAME(PDHDOKA)
```

Response ET:

```
DFS0725I TERMINATE OLREORG COMMAND FOR DB PDHDOKA  COMPLETE. CC=  24
```

Explanation: The TERM OLREORG command is issued for partition PDHDOKA to stop the OLR that is in progress. The command is not successful because OLR is not in progress for the partition.

Example 2 for TERMINATE OLREORG command

TSO SPOC input:

```
TERM OLREORG NAME(PDHDOKA,PDHDOKC)
```

TSO SPOC output:

```
Partition MbrName    CC
    PDHDOKA   IMSA        0
    PDHDOKA   IMS1       24
    PDHDOKC   IMSA       24
    PDHDOKC   IMS1       24
```

OM API input:

```
CMD (    TERM OLREORG NAME(PDHDOKA,PDHDOKC))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.168 21:31:13.035976</statime>
<stotime>2003.168 21:31:13.038227</stotime>
<staseq>B9962C747D6C8868</staseq>
<stoseq>B9962C747DF93586</stoseq>
<rqsttkn1>USRT005 10143113</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
</ctl>
<cmderr>
<mbr name="IMSA    ">
<typ>IMS     </typ>
<styp>DBCTL   </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntext>At least one request successful</rsntext>
</mbr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003004</rsn>
<rsntext>At least one request successful</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMSA    </master>
<userid>USRT005 </userid>
<verb>TERM</verb>
<kwd>OLREORG         </kwd>
<input>TERM OLREORG NAME(PDHDOKA,PDHDOKC) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PART" llbl="Partition" scope="LCL" sort="A" key="1"
 scroll="NO" len="9" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="N" key="0" scroll="NO"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="N" key="0" scroll="YES"
 len="4" dtype="INT" align="right" />
```

```
</cmdrsphdr>
<cmdrspdata>
<rsp> PART(PDHDOKA ) MBR(IMSA    ) CC(   0) </rsp>
<rsp> PART(PDHDOKC ) MBR(IMSA    ) CC(  24) </rsp>
<rsp> PART(PDHDOKA ) MBR(IMS1    ) CC(  24) </rsp>
<rsp> PART(PDHDOKC ) MBR(IMS1    ) CC(  24) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The TERM OLREORG command is issued to stop the OLR for partitions PDHDOKA and PDHDOKC. The command is routed to IMSA and IMS1. The command is successful for partition PDHDOKA at IMSA, where OLR is in progress. The command is not successful for PDHDOKC at IMSA because OLR is not in progress for PDHDOKC on IMSA. The command is not successful for either of the two partitions at IMS1, because OLR is not in progress for PDHDOKA and PDHDOKC on IMS1.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

Chapter 26. /TEST command

Use the /TEST command to place a terminal or a user into either test mode or MFSTEST mode.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 582
- "Examples" on page 583

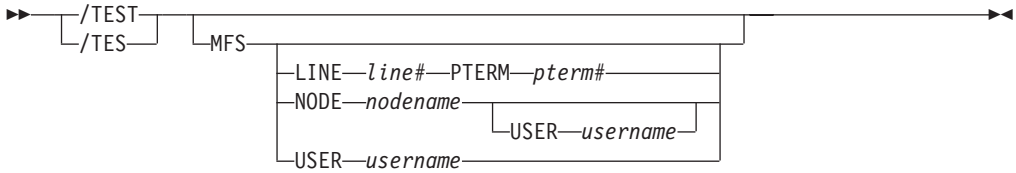
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 286. Valid environments for the /TEST command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TEST	X		X
LINE	X		X
NODE	X		X
PTERM	X		X
USER	X		X

Syntax



Keywords

The following keywords are valid for the /TEST command:

MFS
Specifies MFS test mode for the terminal or user. The MFS parameter is valid only for terminals supported by Message Format Service (MFS). When the /TEST MFS USER command is issued for a dynamic user, it is not possible to determine if the MFSTEST mode is valid until the user signs on to a terminal. Once the user signs on to a terminal, a check is made to determine whether that terminal supports MFSTEST mode. If the terminal does not support MFSTEST mode, the mode is not propagated to the terminal and is removed from the USER unless another /TEST MFS USER command is issued.

© Copyright IBM Corp. 1974, 2015

LINE, PTERM
Specifies the non-VTAM line and pterm to place into MFSTEST mode.

NODE
Specifies the VTAM node to place into MFSTEST mode.

The /TEST MFS NODE form of the command is valid only for statically defined nodes.

Restrictions for using NODE and USER parameters together:

- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- /TEST MFS NODE USER commands are valid for ISC, LUP, and 3600 nodes only. For ISC, the /TEST MFS NODE *nodename* USER *username* form of the command is supported for ISC nodes and applies to the half-session allocated to the USER username.

USER
When specified without the NODE keyword, USER specifies the dynamic user to place into MFSTEST mode. MFSTEST mode status is kept from one signon to another. For example, if a user issues a /TEST MFS command, signs off, and then signs on again at another terminal, the MFSTEST mode is still in effect. If the user does not exist, it is created and MFSTEST mode is set.

Usage notes

This command can be issued to an IMSplex using the Batch SPOC utility.

In test or echo mode, any input from the terminal is sent back. Input and output errors are not checked, and IMS error notification procedures are bypassed. Echo mode continues until reset with an /END, /IAM, /STOP LINE, /STOP LINE PTERM, or a /STOP NODE command. The /DISPLAY command identifies a terminal or user in test mode or MFSTEST mode. If no keywords are supplied, the terminal entering the command is placed into test mode or MFSTEST mode.

Test mode is not a command significant status, so the commands to set test mode are not recoverable nor are they kept after signons and can only be set by the end user or terminal, not remotely by an operator. The /TEST LINE, /TEST NODE, and /TEST USER commands, which set test mode remotely, are no longer supported.

In MFSTEST mode, terminals supported by the Message Format Service use format blocks from a special test library if the requested format block is in the test library; otherwise, the blocks are obtained from the production library. MFSTEST mode continues until reset with an /END command. Certain error conditions can occur that cause MFSTEST mode to terminate. If an error condition occurs, the terminal operator receives an error message.

MFSTEST mode is a command significant status, is recoverable and is remembered across logons and signons. For example, if a /TEST MFS NODE command is entered at a node, the node logs off and logs back on at another terminal, MFSTEST mode is still in effect. If a dynamic user issues a /TEST MFS USER command, signs off, and then signs on again at another terminal, MFSTEST mode is still in effect.

/TEST MFS NODE applies to dynamic nodes in addition to static nodes because MFSTEST mode is associated with dynamic nodes as well as dynamic users. /TEST MFS NODE and /TEST MFS NODE USER set MFSTEST mode at the node level. /TEST MFS USER sets MFSTEST at the user level. /TEST MFS with no keywords sets MFSTEST at the node level for static terminals (they have no user level) and at the user level for dynamic terminals.

The /TEST NODE USER command is supported for static and dynamic ISC sessions. For ISC, /TEST MFS NODE USER is required. You cannot use /TEST MFS NODE (without USER) for ISC and have it apply to all of the half-sessions.

If global resource information is kept in Resource Manager, MFSTEST mode is set globally. If global resource information is not kept in Resource Manager, the resource does not exist, and ETO is enabled, the resource (node or user) is created and MFSTEST mode is set. If a temporary node is dynamically created to hold command status, and the temporary node has MFSTEST status, then, when a logon occurs for the node, the MFSTEST status is set for the logged-on node. If the node logging on is an ISC parallel session, MFSTEST is set only for the first half-session that is logged on. Subsequent ISC half-sessions will not be put into MFSTEST mode.

Examples

The following are examples of the /TEST command:

Example 1 for /TEST command

Entry ET:

```
/TEST
```

Response ET:

```
DFS058I  TEST COMMAND COMPLETED
```

Entry ET:

```
NOW IS THE TIME TO COME TO THE AID
```

Response ET:

```
NOW IS THE TIME TO COME TO THE AID
```

Explanation: The entering terminal is placed in echo mode and continues to receive message input as output until test mode is terminated.

Example 2 for /TEST command

Entry ET:

```
/TEST MFS
```

Response ET:

```
DFS058I  TEST COMMAND COMPLETED
```

Explanation: The entering terminal is placed into MFSTEST mode.

Chapter 27. /TRACE commands

Use the /TRACE commands to direct and to control the IMS capabilities for tracing internal IMS events. They also start, stop, and define the activity to be monitored by the IMS Monitor.

The information developed by the LINE, LINK, NODE, UNITYPE, TRANSACTION, PROGRAM, PSB, and TCO keywords is written on the IMS system log (type X'5F' for PSB, and type X'67' records for the other keywords mentioned.) PI (program isolation) and TABLE trace information is kept in storage or logged (type X'67' records), depending on specification of additional keywords. The MONITOR keyword provides no such output on the system log; it only controls the IMS Monitor. The monitor develops its own output data and writes it on a separate data set.

This command can be issued to an IMSplex using the Batch SPOC utility.

The status and options of the current IMS traces can be displayed with the /DISPLAY TRACE command.

Subsections:

- "/TRACE EXIT command" on page 586
- "/TRACE LINE command" on page 587
- "/TRACE LINK command" on page 589
- "/TRACE LUNAME command" on page 592
- "/TRACE MONITOR command" on page 593
- "/TRACE NODE command" on page 596
- "/TRACE OSAMGTF command" on page 600
- "/TRACE PI command" on page 600
- "/TRACE PGM command" on page 603
- "/TRACE PSB command" on page 605
- "/TRACE TABLE command" on page 606
- "/TRACE TCO command" on page 610
- "/TRACE TIMEOUT command" on page 611
- "/TRACE TMEMBER command" on page 612
- "/TRACE TRAN command" on page 614
- "/TRACE TRAP command" on page 615
- "/TRACE UNITYPE command" on page 616

Related concepts:

Using IMS reports (System Administration)

Related reference:

"UPDATE PGM command" on page 755

© Copyright IBM Corp. 1974, 2015

/TRACE EXIT command

Use the /TRACE EXIT command to identify user exit tracing.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"

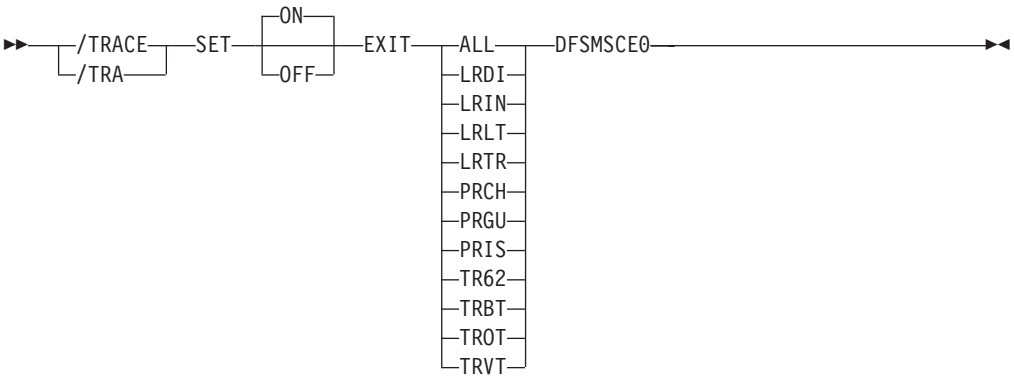
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 287. Valid environments for the /TRACE EXIT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
EXIT	X		X
SET	X	X	X

Syntax



Keywords

The following keyword is valid for the /TRACE EXIT command:

DFSMSCE0
The TM and MSC Message Routing and Control user exit. When this keyword is used on the /TRACE EXIT command, the traces causes IMS to write type 6701-MSEA and 6701-MSEB log records to the log data set when the exit routine is called. A 6701-MSEA record is logged when the exit is called if the trace is active for the entry point. A 6701-MSEB record is logged when the exit returns to IMS.

Usage notes

When using the EXIT keyword, you must specify one of the following parameters to turn on or off:

ALL The trace is turned on or off for all entry points.

LRDI Calls the Link Receive Direct Routing exit entry point.

LRIN Calls the Link Receive Intermediate exit entry point.

LRLT Calls the Link Receive LTERM exit entry point.

LRTR Calls the Link Receive Transaction exit entry point.

PRCH Calls the Program Routing CHNG Call exit entry point.

PRGU Calls the Program Routing GU Call exit entry point.

PRIS Calls the Program Routing ISRT Call exit entry point.

TR62 Calls the Terminal Routing LU62 exit entry point.

TRBT Calls the Terminal Routing non-VTAM exit entry point.

TROT Calls the Terminal Routing OTMA exit entry point.

TRVT Calls the Terminal Routing VTAM exit entry point.

/TRACE LINE command

Use the /TRACE LINE command to cause events related to the lines to be traced.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 588
- "Examples" on page 589

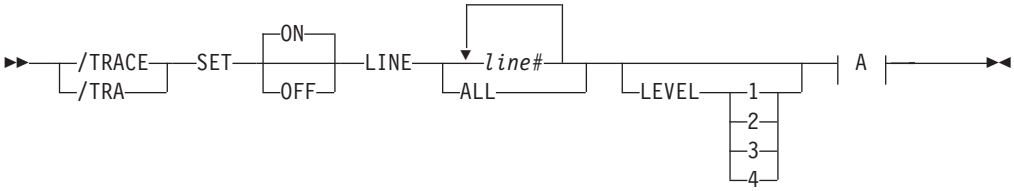
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 288. Valid environments for the /TRACE LINE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
LEVEL	X		X
LINE	X		X
MODULE	X		X
SET	X	X	X
TAKEOVER	X		X

Syntax



A:

Keywords

The following keywords are valid for the /TRACE LINE command:

LEVEL
Expands the LINE, LINK, NODE, or UNITYPE trace functions. The LEVEL specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

LEVEL indicates the extent of the control block trace information desired. The indicated control blocks are only traced at relevant times. All levels are inclusive of numerically lower levels. The following list displays the levels and their associated blocks.

Level	Blocks
1	CLB (DECB) or LLB(MSC)
	CTB or LTB(MSC)
	IOB (for non-VTAM lines) or IOSB (MSC for channel-to-channel links)
2	CNT or LNB(MSC)
	CXB
	CRB
	CIB
	CCB
	PD stack
3	queue manager buffers
	Input/output line buffers
	LXB (for channel-to-channel links and processor storage to processor storage)
4	save area sets (IMS dispatching)

If the first /TRACE SET ON command does not specify LEVEL, a default of 4 will be used. Specifying LEVEL on subsequent commands will change the defaults.

MODULE
Is used to expand the LINE, LINK, NODE, or UNITYPE trace functions. The MODULE specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

MODULE indicates which modules are to have their control blocks traced.

ALL Both device-dependent module (DDM) and MFS

DDM Communication analyzer and device-dependent module interfaces

MFS Communication analyzer and Message Format Service (MFS) module interfaces

If the first /TRACE SET ON command does not specify MODULE, a default of ALL will be used. Specifying MODULE on subsequent commands will change the defaults.

TAKEOVER

Controls tracing during takeover only, and is separate from regular tracing. TAKEOVER tracing can be set for LINE, LINK, NODE, and UNITYPE keywords.

TAKEOVER only applies in an XRF environment. When TAKEOVER is used with SET OFF, the trace is turned off before takeover. When an output message is dequeued for a terminal, takeover tracing will stop for that terminal.

If both regular and takeover tracing are entered, the most recent setting will override any previous settings; for example, takeover tracing will override regular tracing if regular tracing was entered first. This means that either regular or TAKEOVER tracing can be in effect, but not both.

/TRACE TAKEOVER can be issued only from an XRF active system. It is rejected if entered from an alternate or non-XRF system. It is recovered across restart and takeover, and only needs to be entered once until cold start. Tracing occurs only if the session was active at the time of the takeover.

Examples

The following is an example of the /TRACE LINE command:

To turn on the Message Format Service module and communication analyzer level 4 control block tracing for all physical terminals on line 4.

Entry ET:

```
/TRACE SET ON LINE 4 LEVEL 4 MODULE MFS
```

Response ET:

```
DFS058I  TRACE COMMAND COMPLETED
```

/TRACE LINK command

Use the /TRACE LINK command to cause events related to the logical links to be traced.

Subsections:

- "Environment"
- "Syntax" on page 590
- "Keywords" on page 590
- "Examples" on page 591

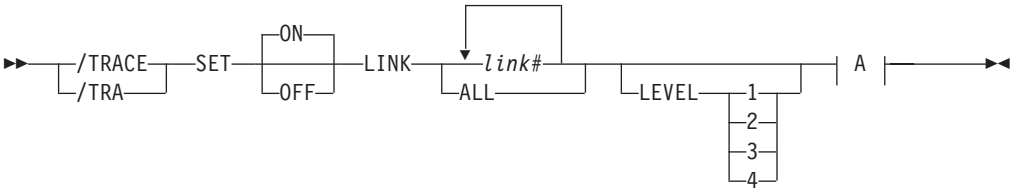
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 289. Valid environments for the /TRACE LINK command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
LEVEL	X		X
LINK	X		X
MODULE	X		X
SET	X	X	X
TAKEOVER	X		X

Syntax



A:

Keywords

The following keywords are valid for the /TRACE LINK command:

LEVEL

Expands the LINE, LINK, NODE, or UNITYPE trace functions. The LEVEL specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

LEVEL indicates the extent of the control block trace information desired. The indicated control blocks are only traced at relevant times. All levels are inclusive of numerically lower levels. The following list displays the levels and their associated blocks.

Level	Blocks
1	CLB (DECB) or LLB(MSC)
	CTB or LTB(MSC)
	IOB (for non-VTAM lines) or IOSB (MSC for channel-to-channel links)
2	CNT or LNB(MSC)
	CXB
	CRB
	CIB
	CCB

PD stack

3 queue manager buffers

Input/output line buffers

LXB (for channel-to-channel links and processor storage to processor storage)

4 save area sets (IMS dispatching)

If the first /TRACE SET ON command does not specify LEVEL, a default of 4 will be used. Specifying LEVEL on subsequent commands will change the defaults.

MODULE
Is used to expand the LINE, LINK, NODE, or UNITYPE trace functions. The MODULE specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

MODULE indicates which modules are to have their control blocks traced.

ALL Both device-dependent module (DDM) and MFS

DDM Communication analyzer and device-dependent module interfaces

MFS Communication analyzer and Message Format Service module interfaces

If the first /TRACE SET ON command does not specify MODULE, a default of ALL will be used. Specifying MODULE on subsequent commands will change the defaults.

TAKEOVER
Controls tracing during takeover only, and is separate from regular tracing. TAKEOVER tracing can be set for LINE, LINK, NODE, and UNITYPE keywords.

TAKEOVER only applies in an XRF environment. When TAKEOVER is used with SET OFF, the trace is turned off before takeover. When an output message is dequeued for a terminal, takeover tracing will stop for that terminal.

If both regular and takeover tracing are entered, the most recent setting will override any previous settings; for example, takeover tracing will override regular tracing if regular tracing was entered first. This means that either regular or TAKEOVER tracing can be in effect, but not both.

/TRACE TAKEOVER can be issued only from an XRF active system. It is rejected if entered from an alternate or non-XRF system. It is recovered across restart and takeover, and only needs to be entered once until cold start. Tracing occurs only if the session was active at the time of the takeover.

Examples

The following is an example of the /TRACE LINK command:

To turn on tracing for a logical link:

Entry ET:

```
/TRACE SET ON LINK 2 LEVEL 4 MODULE ALL
```

Response ET:

```
DFS058I  TRACE COMMAND COMPLETED
```

/TRACE LUNAME command

Use the /TRACE LUNAME command to activate and deactivate tracing for a particular LU name or TP name of the LU name.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"

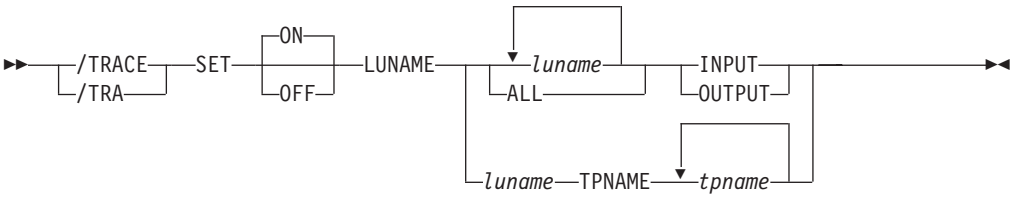
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 290. Valid environments for the /TRACE LUNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
INPUT	X		X
LUNAME	X		X
OUTPUT	X		X
SET	X	X	X

Syntax



Usage notes

Specifying this command causes trace entries to be written to the LUMI trace table. For this reason, the /TRACE SET ON TABLE LUMI command must be entered first in order to create the table for trace entries that will be created by subsequent /TRACE SET ON LUNAME commands. A trace entry is written:

- On LU 6.2 module entries/exits
- When APPC calls are made
- When errors are encountered

The INPUT and OUTPUT keywords provide the operator with the flexibility to control the volume of trace data for LU 6.2 devices.

Specifying the keyword INPUT with the LUNAME keyword indicates tracing is activated or deactivated for input and synchronous outbound activities. Specifying

the parameter ALL with the INPUT keyword causes all future LU 6.2 input and synchronous outbound activities to be traced as well.

Specifying the keyword OUTPUT with the LUNAME keyword indicates tracing is activated or deactivated for asynchronous outbound activities. Specifying the parameter ALL with OUTPUT causes all future LU 6.2 asynchronous outbound activities to be traced as well.

Specifying neither INPUT or OUTPUT is the same as both INPUT and Tracing is activated or deactivated for input and both synchronous and asynchronous outbound activities. Specifying the parameter ALL in this case causes all future LU 6.2 inbound activities, synchronous and asynchronous outbound activities to be traced as well.

The network-qualified LU name is optional for the LUNAME keyword. If the LU name is not a network-qualified LU name and no TP name is specified, tracing is activated or deactivated for all the network-qualified LU names in the system whose LU name matches the LU name specified.

If the specified resource does not exist and tracing is activated, a structure is created to retain the status.

/TRACE MONITOR command

Use the /TRACE MONITOR command to activate or deactivate the IMS Monitor.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 594
- "Usage notes" on page 595
- "Examples" on page 596

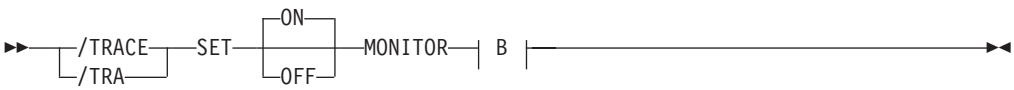
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

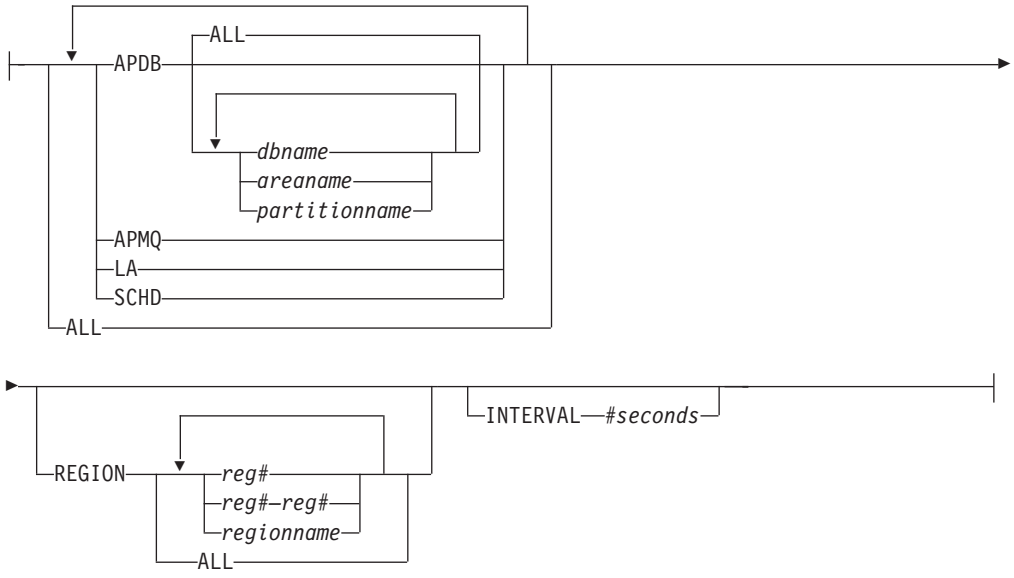
Table 291. Valid environments for the /TRACE MONITOR command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
MONITOR	X	X	X
SET	X	X	X

Syntax



B:



Keywords

When activating the monitor, you must specify one or more of the following MONITOR parameters to indicate the events to be monitored:

ALL
Monitors all of the activity in this list.

APDB
Monitors activity between application programs and databases, including Fast Path activity. Monitoring includes all application program requests to external subsystem databases. Monitoring can optionally be limited to a subset of full-function databases or partitions, Fast Path DEDBs and MSDBs, and the areas comprising those DEDBs by specifying database names or area names.

You can specify the ALL parameter to indicate all databases, areas, and partitions, or you can explicitly enter database names, area names, and partition names.

APMQ
Monitors activity between application programs and message queues, including Fast Path activity.

INTERVAL
Monitors events for a fixed interval of time, entered in seconds. INTERVAL defines the period of time after which no monitor log records will be written.

The duration of the monitoring must be less than twenty-four hours (86,400 seconds). When INTERVAL is not specified, monitoring will continue until the /TRACE SET MONITOR OFF command is issued, or until IMS shuts down.

INTERVAL does not define when the Monitor will be turned off, because the IMS Monitor will not be turned off until the first attempt is made to write a monitor log record after the defined interval has expired.

LA Monitors line and logical link events.

REGION
Monitors events related to specific dependent regions. The regions might or might not currently be active. Each region can be specified as:

- A region number from 1 to 999 (*reg#*). The number cannot exceed the MAXPST with which IMS was brought up.
- A range of region numbers from 1 to 999 (*reg#–reg#*). The number cannot exceed the MAXPST with which IMS was brought up.
- A region name (*regionname*).

If REGION is not specified, or when REGION ALL is specified, the activities of all dependent regions are monitored.

SCHD
Monitors scheduling and termination events, including Fast Path activities.

Usage notes

You can specify any combination of ALL, APDB, APMQ, INTERVAL, LA, REGION, and SCHD parameters on the MONITOR keyword, as shown in the following table.

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the MONITOR keyword parameters can be issued.

Table 292. MONITOR keyword parameter environments

Keyword parameter	DB/DC	DBCTL	DCCTL
ALL	X	X	X
APDB	X	X	
APMQ	X		X
INTERVAL	X	X	X
LA	X		X
REGION	X	X	X
SCHD	X	X	X

The IMS Monitor report output varies depending upon which keywords or parameters you specify. Sections of the report can be misleading if required records are excluded. For example, if you specify the APDB parameter without the SCHD parameter, PSB/PCB relationships will not be correctly represented. To get the total DL/I call reports without the IMS line activity, the correct parameters to specify are APDB, APMQ, and SCHD.

The monitor writes log records until one of the following occurs:

- /TRACE SET OFF MONITOR is entered.
- The time interval specified by the INTERVAL parameter is reached.
- IMS is shut down.

When deactivating the monitor, no parameters are required. Any parameters that are entered, other than ALL, are ignored. ALL is the default. If the monitor is to be reactivated, a new set of MONITOR keyword parameters must be selected.

Examples

The following are examples of the /TRACE MONITOR command:

Entry ET:

```
/TRACE SET ON MONITOR ALL
```

Response ET:

```
DFS058I TRACE COMMAND COMPLETED
DFS2500I DATASET DFSDCMON SUCCESSFULLY ALLOCATED
DFS2212I DC MONITOR STARTED
```

Explanation: The monitor is activated and all events will be monitored.

Entry ET:

```
/TRACE SET OFF MONITOR
```

Response ET:

```
DFS058I TRACE COMMAND COMPLETED
DFS2500I DATASET DFSDCMON SUCCESSFULY DEALLOCATED
DFS2212I DC MONITOR STOPPED
```

Explanation: The monitor is deactivated.

Entry ET:

```
/TRACE SET ON MONITOR LA
```

Response ET:

```
DFS058I TRACE COMMAND COMPLETED
```

Explanation: The monitor is activated. Line and logical link activities will be monitored. The monitoring of events from the previous activation of the monitor no longer apply.

/TRACE NODE command

Use the /TRACE NODE command to cause events related to the node or nodes to be traced.

Subsections:

- "Environment"
- "Syntax" on page 597
- "Keywords" on page 597
- "Usage notes" on page 598
- "Examples" on page 599

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

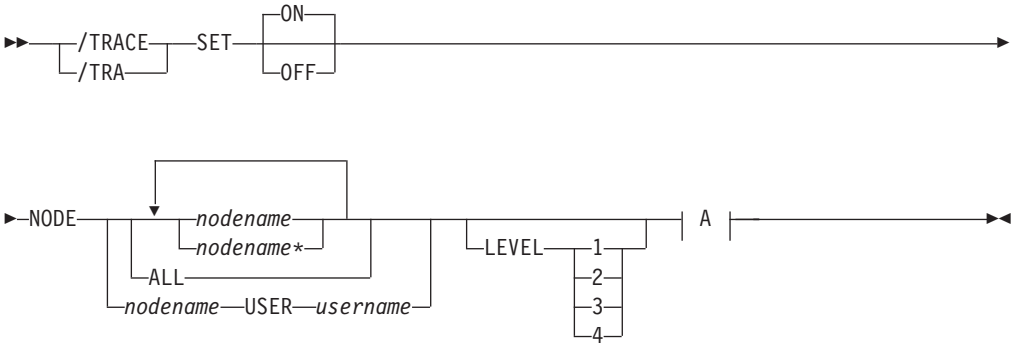
Table 293. Valid environments for the /TRACE NODE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X

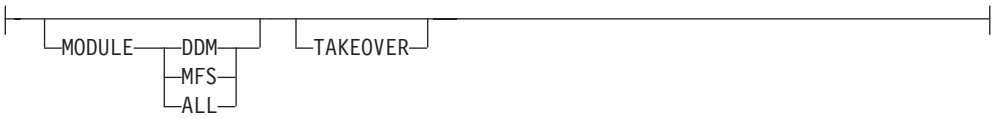
Table 293. Valid environments for the /TRACE NODE command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
LEVEL	X		X
MODULE	X		X
NODE	X		X
SET	X	X	X
TAKEOVER	X		X
USER	X		X

Syntax



A:



Keywords

The following keywords are valid for the /TRACE NODE command:

LEVEL

Expands the LINE, LINK, NODE, or UNITYPE trace functions. The LEVEL specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

LEVEL indicates the extent of the control block trace information desired. The indicated control blocks are only traced at relevant times. All levels are inclusive of numerically lower levels. The following list displays the levels and their associated blocks.

Level	Blocks
1	CLB (DECB) or LLB(MSC)
	CTB or LTB(MSC)
	IOB (for non-VTAM lines) or IOSB (MSC for channel-to-channel links)
2	CNT or LNB(MSC)
	CXB

CRB

CIB

CCB

PD stack

3 queue manager buffers

Input/output line buffers

LXB (for channel-to-channel links and processor storage to processor storage)

4 save area sets (IMS dispatching)

If the first /TRACE SET ON command does not specify LEVEL, a default of 4 will be used. Specifying LEVEL on subsequent commands will change the defaults.

MODULE
Is used to expand the LINE, LINK, NODE, or UNITYPE trace functions. The MODULE specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

MODULE indicates which modules are to have their control blocks traced.

ALL Both device-dependent module (DDM) and MFS

DDM Communication analyzer and device-dependent module interfaces

MFS Communication analyzer and Message Format Service module interfaces

If the first /TRACE SET ON command does not specify MODULE, a default of ALL will be used. Specifying MODULE on subsequent commands will change the defaults.

TAKEOVER
Controls tracing during takeover only, and is separate from regular tracing. TAKEOVER tracing can be set for LINE, LINK, NODE, and UNITYPE keywords.

TAKEOVER only applies in an XRF environment. When TAKEOVER is used with SET OFF, the trace is turned off before takeover. When an output message is dequeued for a terminal, takeover tracing will stop for that terminal.

If both regular and takeover tracing are entered, the most recent setting will override any previous settings; for example, takeover tracing will override regular tracing if regular tracing was entered first. This means that either regular or TAKEOVER tracing can be in effect, but not both.

/TRACE TAKEOVER can be issued only from an XRF active system. It is rejected if entered from an alternate or non-XRF system. It is recovered across restart and takeover, and only needs to be entered once until cold start. Tracing occurs only if the session was active at the time of the takeover.

Usage notes

The NODE parameter can be generic if the USER keyword is not specified and applies to nodes that already exist. Generic NODE parameters do not cause any dynamic nodes to be created.

For ISC nodes, the /TRACE NODE nodename without the USER applies to all half-sessions for NODE nodename, including dynamic ISC sessions that are dynamically allocated later.

For nodes that do not exist, /TRACE SET ON NODE *nodename* without the USER keyword causes the dynamic NODE nodename to be created to maintain knowledge of the trace request when the node becomes active. Until the node becomes active, /DISPLAY NODE shows a type of UNK (unknown). If the trace is subsequently turned off and the temporary node still exists, it is deleted at the next checkpoint.

Restrictions for using NODE and USER parameters together:

- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- /TRACE NODE USER commands are valid for ISC and non-ISC nodes and users.

If global resource information is kept in Resource Manager, /TRACE NODE sets a global trace status for the node and sets the trace status locally. If global resource information is not kept in Resource Manager, /TRACE NODE sets the trace status locally. If the node does not exist in an ETO environment, IMS creates the node and sets trace status for the local node.

Examples

The following are examples of the /TRACE NODE command:

The following example shows how IMS creates temporary nodes to retain trace status data.

The /TRACE command is issued for a dynamic ISC NODE that does not yet exist, DTSLU607, causing a temporary node to be created to retain the trace status. Once the ISC parallel sessions IMSUS01 and IMSUS02 are allocated, the trace status is applied to them both.

Entry ET:

```
/TRACE SET ON NODE DTSLU607
```

Response ET:

```
DFS058 TRACE COMMAND COMPLETED
```

Entry ET:

```
/DISPLAY NODE DTSLU607
```

Response ET:

```
NODE-USR TYPE   CID       RECD ENQCT DEQCT   QCT  SENT
DTSLU607 UNK    00000000     0     0     0     0     0 TRA
*90127/091634*
```

Entry ET:

```
/DISPLAY NODE DTSLU607
```

Response ET:

```
NODE-USR TYPE   CID       RECD ENQCT DEQCT   QCT  SENT
DTSLU607 LUT6
-N/A     UNK    00000000     0     0     0     0     0 TRA
-IMSUS01        01000002     0     0     0     0     0 SIGN(IMSUS01 )
                                                       IDLE CON TRA PRI
-IMSUS02        01000004     0     0     0     0     0 SIGN(IMSUS02 )
                                                       IDLE CON TRA PRI
*90127/091432*
```

/TRACE OSAMGTF command

Use the /TRACE SET OSAMGTF command to activate or deactivate the OSAM Buffer Handle GTF (Generalized Trace Facility) subroutine. When activated, the OSAMGTF trace generates the z/OS Generalized Trace Facility trace entries for all the IMS requests to the OSAM Buffer Handler.

Subsections:

- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 294. Valid environments for the /TRACE OSAMGTF command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
OSAMGTF	X	X	
SET	X	X	X

Syntax

/TRACE PI command

Use the /TRACE PI command to cause program isolation trace entries to be written to a trace table.

Subsections:

- "Environment" on page 601
- "Syntax" on page 601
- "Keywords" on page 601
- "Usage notes" on page 602
- "Examples" on page 602

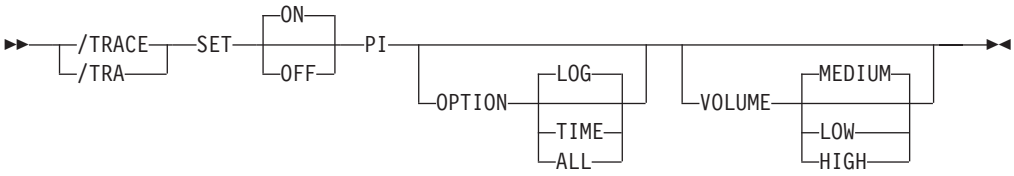
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 295. Valid environments for the /TRACE PI command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
OPTION	X	X	X
PI	X	X	
SET	X	X	X
VOLUME	X	X	X

Syntax



Keywords

The following keywords are valid for the /TRACE PI command:

OPTION
Indicates a request for one of the following program isolation trace options:

ALL Both LOG and TIME.

LOG Requests that traced data be written to the system log. If SET OFF, tracing continues but no buffers are transferred to the system log. LOG is the default.

The log option includes the possibility of externally tracing to a data set other than the IMS OLDS. If specified, DASD external tracing has first priority, TAPE external tracing has second priority, and IMS OLDS has third priority. External tracing to the OLDS is not done without operator approval. EXTERNAL trace is available to the alternate system only for DASD/TAPE type, but not for OLDS.

TIME Requests that an additional time field be included in each ENQ/DEQ request trace record if a WAIT was needed. This field will contain elapsed wait time. If set OFF, tracing continues but only the time of day is recorded.

Entries for Fast Path have no elapsed wait time.

VOLUME
Specifies the volume of entries to be written to the PI trace table: LOW volume, MEDIUM volume (default), or HIGH volume.

Usage notes

PI trace entries are written in the same trace table as DL/I and lock activity trace entries. A PI trace entry contains information about program isolation ENQ/DEQ calls and DL/I calls. The trace entry created by /TRACE TABLE DLI contains different information about DL/I calls and is written as a separate entry in the same trace table. Starting the LOCK trace also causes PI tracing to occur.

If PI is entered without the OPTION keyword, the program isolation trace is kept in storage without being logged. If you are using the program isolation trace to provide statistics and performance data, you should enter OPTION(ALL).

The following table lists various /TRACE command formats and shows whether the command influences tracing, logging, and the additional time field.

Table 296. /TRACE command formats

Command	Tracing	Logging	Additional time field
/TRACE SET ON PI	Yes	No	No
/TRACE SET OFF PI	No	No	No
/TRACE SET ON PI OPTION[1]	Yes	Yes	No
/TRACE SET OFF PI OPTION[1]	Yes	No	No
/TRACE SET ON PI OPTION TIME	Yes	No	Yes
/TRACE SET OFF PI OPTION TIME	Yes	No	No
/TRACE SET ON PI OPTION ALL	Yes	Yes	Yes
/TRACE SET OFF PI OPTION ALL	Yes	No	No

Note:

1. This is the same command as /TRACE SET ON | OFF PI OPTION LOG.

Examples

The following are examples of the /TRACE PI command:

Example 1

To turn on program isolation tracing, include the additional time field in the trace record and have the trace information logged:

Entry ET:

```
/TRACE SET PI OPTION ALL
```

Response ET:

```
DFS058I  TRACE COMMAND COMPLETED
```

Example 2

To turn off logging of program isolation trace data but continue the trace in storage:

Entry ET:

```
/TRACE SET OFF PI OPTION LOG
```

Response ET:

```
DFS058I  TRACE COMMAND COMPLETED
```

Example 3

To stop program isolation tracing.

Entry ET:

```
/TRACE SET OFF PI
```

Response ET:

```
DFS058I  TRACE COMMAND COMPLETED
```

/TRACE PGM command

Use the /TRACE PGM command to trace the DL/I portion of Data Communications (DC) for a specific program.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands" on page 604

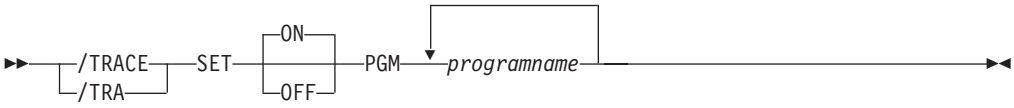
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 297. Valid environments for the /TRACE PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
PGM	X	X	X
SET	X	X	X

Syntax



Usage notes

Each DL/I call to a TPPCB, issued by the user application program, is traced on entry to and exit from the DC call handler DFSDLA30. On entry to DFSDLA30 a type 6701-LA3A record is written, on exit from DFSDLA30 a type 6701-LA3B record is written.

Each record will contain the following items if applicable:

- TPPCB
- Up to 64 bytes of the I/O area
- SMB
- PST

If the batch message program (BMP) being traced is IBM IMS Queue Control Facility for z/OS, a 6701-MRQB record is logged by the IMS Queue Control Facility module DFSQMRQ0. The default program name for the IMS Queue Control Facility BMP is MRQPSB, and can be overridden on the MSGQUEUE system definition macro.

Items logged in the 6701-MRQB record, if applicable, are:

- TPPCB
- AIB
- I/O AREA
- PST
- QTPDST
- QSAPWKAD
- QMBA
- PSTDCA
- REG14-12

When CPI Communications driven transaction programs issue the DL/I APSB call specifying a PSB that contains alternate PCBs, only the PGM keyword is applicable.

Equivalent IMS type-2 commands

The following table shows variations of the /TRACE PGM command and the IMS type-2 commands that perform similar functions.

Table 298. Type-2 equivalents for the /TRACE PGM command

Task	/TRACE PGM command	Similar IMS type-2 command
Starts the tracing of a program.	/TRACE SET ON PGM *pgmname*	UPDATE PGM NAME(*pgmname*) START(TRACE)
Stops the tracing of a program.	/TRACE SET OFF PGM *pgmname*	UPDATE PGM NAME(*pgmname*) STOP(TRACE)

Related concepts:

Diagnosing problems in the Queue Control Facility Message Requeuer (Diagnosis)

Related reference:

IMS Queue Control Facility overview

"UPDATE PGM command" on page 755

/TRACE PSB command

Use the /TRACE PSB command to record all full function IMS DL/I database calls issued for the named PSB. FP/DC/SAA calls are not captured when /TRACE SET ON PSB initiated tracing of PSBs.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Examples" on page 606

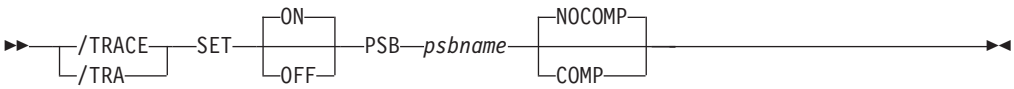
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 299. Valid environments for the /TRACE PSB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
COMP	X	X	
NOCOMP	X	X	
PSB	x	X	
SET	X	X	X

Syntax



Keywords

The following keywords are valid for the /TRACE PSB command:

COMP
Used with the /TRACE SET PSB command to generate PCB and data-compare statement images.

The /TRACE SET PSB *psbname* COMP command only applies to BMPs in a DBCTL environment.

NOCOMP
Prevents PCB and data-compare statement images from being generated. NOCOMP is the default.

Usage notes

For LU 6.2, the PSB keyword is applicable only if the CPI Communications driven transaction program has issued a DL/I APSB call to allocate a PSB.

The information resulting from the use of this keyword is written on the X'5F' log record.

Examples

The following is an example of the /TRACE PSB command:

To trace all DL/I calls issued for PSB AALST:

Entry ET:

```
/TRACE SET ON PSB AALST COMP
```

Response ET:

```
DFS058I  TRACE COMMAND COMPLETED
```

/TRACE TABLE command

Use the /TRACE TABLE command with the SET keyword to start or stop online tracing into the specified trace tables.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 609
- "Examples" on page 609

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 300. Valid environments for the /TRACE TABLE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
OPTION	X		X
SET	X	X	X
TABLE	X	X	X
VOLUME	X	X	X

Syntax

Keywords

The following keywords are valid for the /TRACE TABLE command:

OPTION
Indicates whether the trace tables are to be logged.

NOLOG
Indicates that the trace tables are not logged. This means that there will be no record of trace tables that have been reused. NOLOG is the default.

The same trace table is used for DL/I, LOCK, and PI trace information. Whether this table is logged depends on the logging option of the last /TRACE command that was entered for DL/I, PI, or LOCK.

LOG
Requests that the trace tables be written to the system log before they are reused (wrapped).

The log option includes the possibility of externally tracing to a data set other than the IMS OLDS. If specified, DASD external tracing has first priority, TAPE external tracing has second priority, and IMS OLDS has third priority. External tracing to the OLDS is not done without operator approval. This option is ignored for the retrieve trace and the Fast Path trace (default).

FPTRACE has its own SYSOUT. External trace is available to the alternate system only for DASD/TAPE type, but not for OLDS.

VOLUME
Specifies the volume of entries to be written to the specified trace table: LOW volume, MEDIUM volume (default), or HIGH volume.

The VOLUME keyword is valid on active and RSR tracking subsystems.

You can specify the trace that you want to activate or deactivate by using one of the following TABLE keyword parameters:

ALL
Indicates that traces into all trace tables are to be enabled or disabled. This is the default.

CSLT
Indicates that the CSL trace is to be activated or deactivated.

DIAG
Indicates that the /DIAGNOSE command trace tables are to be activated or deactivated.

DISP
Indicates that the dispatcher trace is to be activated or deactivated.

DL/I
Indicates that DL/I tracing is to be activated or deactivated.

DLOG
Indicates that the logging trace is to be activated or deactivated.

FAST
Indicates that the Fast Path trace is to be activated or deactivated.

Fast Path trace is activated by including the FPTRACE DD statement in the dependent region JCL to define the destination of the trace output and by issuing the operator command /TRACE SET ON TABLE FAST.

Recommendation: Run this trace only in a test environment because the FPTRACE output is large.

FPTT
Indicates that the Fast Path table trace is to be activated or deactivated.

IDC0
Indicates that tracing of errors in modules DFSCNXA0 and DFSIDC00 is to be activated or deactivated.

LATC
Indicates that the latch trace is to be activated or deactivated.

LOCK
Indicates that LOCK and PI tracing is to be activated or deactivated.

LRTT
Indicates that the log router trace table is to be activated or deactivated. LRTT is only valid for an RSR tracking subsystem.

LUMI
Indicates that the LU 6.2 manager trace is to be activated or deactivated. /TRACE SET ON TABLE LUMI must be entered first before entering any /TRACE SET ON LUNAME command in order to create the LUMI trace table for trace entries.

LUMI is not valid for an RSR tracking subsystem.

MSCT
Indicates that the MSC trace is to be activated or deactivated.

OCMD
Indicates that the OM command trace is to be activated or deactivated.

ODBA
Indicates that the ODBA trace table is to be activated or deactivated. ODBA is not valid in an RSR tracker, batch, or DCCTL environment.

ORTT
Indicates that the Online Recovery System trace is to be activated or deactivated.

OTMT
Indicates that the IMS Open Transaction Manager Access (OTMA) trace is to be activated or deactivated.

QMGR
Indicates that the queue manager trace is to be activated or deactivated.

RETR
Indicates that the DL/I retrieve trace is to be activated or deactivated. RETR is not valid for an RSR tracking subsystem.

RRST
Indicates that the Resource Recovery trace is activated or deactivated.

SCHD
Indicates that the scheduler trace is to be activated or deactivated. SCHD is not valid for an RSR tracking subsystem.

SQTT
Indicates that the shared queues trace is to be activated or deactivated. SQTT is only valid in a shared-queues environment.

STRG
Indicates that the storage manager trace is to be activated or deactivated.

SUBS
Indicates that the external subsystem trace is to be activated or deactivated. SUBS is not valid for an RSR tracking subsystem.

Usage notes

The TABLE keyword parameter indicates the specific trace that is to be activated or deactivated. The DL/I, LOCK, and PI traces share the same trace tables. However, turning on the DL/I trace does not turn on the LOCK trace, and vice versa.

The following trace into trace tables can be turned on or off with the online /TRACE command.

The following table shows the environments in which the trace tables are valid.

Table 301. Trace tables and environments in which they are valid

Trace table	DB/DC	DBCTL	DCCTL
ALL	X	X	X
CSLT	X	X	X
DIAG	X	X	X
DISP	X	X	X
DL/I	X	X	X
DLOG	X	X	X
FAST	X	X	X
FPTT	X	X	X
IDC0	X		X
LATC	X	X	X
LOCK	X	X	
LRTT	X	X	X
LUMI	X		X
MSCT	X		X
OCMD	X	X	X
ODBA	X	X	
ORTT	X		X
OTMT	X	X	
QMGR	X		X
RETR	X	X	
RRST	X		X
SCHD	X	X	X
SQTT	X		X
STRG	X	X	X
SUBS	X	X	X

Examples

The following are examples of the /TRACE TABLE command:

Example 1 for /TRACE TABLE command

To turn on online tracing into the DL/I trace table:

Entry ET:

```
/TRACE SET ON TABLE DL/I
```

Response ET:

```
DFS058I TRACE COMMAND COMPLETED
```

Example 2 for /TRACE TABLE command

To turn on the dispatcher's trace tables and have them written to the system log:

Entry ET:

```
/TRACE SET ON TABLE DISP OPTION LOG
```

When the dispatcher's trace tables are no longer required:

Entry ET:

```
/TRACE SET OFF TABLE DISP
```

Response ET:

```
DFS058I  TRACE COMMAND COMPLETED
```

Example 3 for /TRACE TABLE command

To turn on storage manager trace tables:

Entry ET:

```
/TRACE SET ON TABLE STRG
```

Response ET:

```
DFS058I TRACE COMMAND COMPLETED
```

When the storage manager trace tables are no longer needed:

Entry ET:

```
/TRACE SET OFF TABLE STRG
```

Response ET:

```
DFS058I TRACE COMMAND COMPLETED
```

/TRACE TCO command

Use the /TRACE TCO command to trace TCO (Time Controlled Operation) activity. For the first /TRACE command with the TCO keyword, the default module and level information is used.

TCO trace is basically a DC LINE or NODE trace, and the information developed is also written on the type X'67' log record.

Subsections:

- "Environment" on page 611
- "Syntax" on page 611

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 302. Valid environments for the /TRACE TCO command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
SET	X	X	X
TCO	X	X	X

Syntax

/TRACE TIMEOUT command

Use the /TRACE TIMEOUT command to start or stop the I/O Timeout Detection facility.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 612
- "Usage notes" on page 612

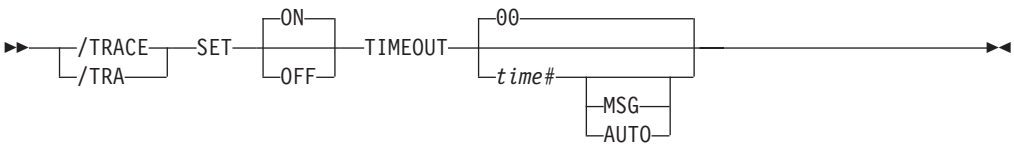
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 303. Valid environments for the /TRACE TIMEOUT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
AUTO	X		X
MSG	X		X
SET	X	X	X
TIMEOUT	X		X

Syntax



Keywords

The following keywords are valid for the /TRACE TIMEOUT command:

time#
is the number of minutes used to determine if the I/O response is overdue. After this number of minutes, time has run out for the response. The range is from 1 through 60; the default value is 0.

If *time#* is 0, or not specified, then when I/O is initiated for a node, the node will be placed on a queue, so that its status can be displayed with the /DISPLAY TIMEOVER command. You will not be notified if the node does not receive a response and the time elapses, and the node will not be reactivated.

If *time#* is not 0, then the following keywords can be used.

MSG
Indicates that a message is issued to the master terminal when I/O takes longer than *time#* minutes. The message indicates that the time has elapsed.

AUTO
IMS issues a message to the master terminal, then perform a VTAM VARY NET,INACT and a VARY NET,ACT, if I/O takes longer than *time#* minutes. An /OPNDST is performed for operable devices that are not shared. For ISC nodes, a message is issued, but there is no automatic (AUTO) restart of any sessions and no VTAM VARY commands issued.

Usage notes

If you want to change the time period or the action to be taken if timeout occurs, you can enter the /TRACE SET ... TIMEOUT command while the Timeout Detection facility is already active. However, if nodes are receiving or sending input or output, they will function according to the previous settings of the /TRACE ... TIMEOUT command. If this is undesirable, then you should enter the /TRACE SET OFF TIMEOUT command before reentering /TRACE SET ... TIMEOUT.

If the timeout trace facility failed during IMS initialization, the /TRACE SET ... TIMEOUT command is rejected with an error message.

The VTAM TIMEOUT I/O facility is automatically started during IMS shutdown. It is set for 1 minute and AUTO.

/TRACE TMEMBER command

Use the /TRACE TMEMBER command to trace IMS Open Transaction Manager Access (OTMA) client activity for OTMA clients.

Subsections:

- "Environment" on page 613
- "Syntax" on page 613
- "Keywords" on page 613
- "Usage notes" on page 613
- "Examples" on page 614

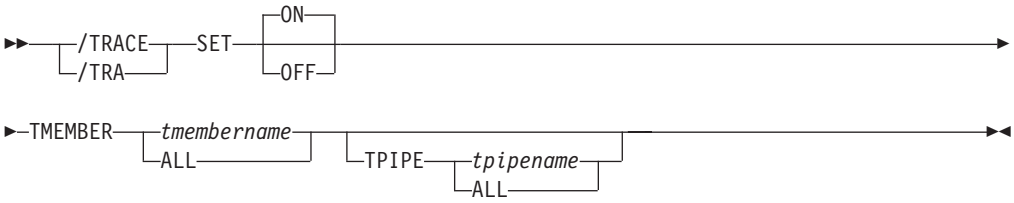
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 304. Valid environments for the /TRACE TMEMBER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
SET	X	X	X
TMEMBER	X		X
TPIPE	x		X

Syntax



Keywords

The following keyword is valid for the /TRACE TMEMBER command:

TPIPE
Is used to trace transaction pipe activity for OTMA clients.

While processing the /TRACE TMEMBER TPIPE command, IMS creates a temporary transaction pipe (if one does not already exist) with the trace status. IMS sets the synchronization status for this transaction pipe when it sends or receives the first message for the transaction pipe.

If the member specified is a super member, trace status is updated for the super member's transaction pipe. If the member specified is a regular member whose hold queue output is managed by a super member, trace status is updated for both the regular member's transaction pipe and the super member's transaction pipe.

Usage notes

If the member specified is a super member, trace status is updated for the super member. If the member specified is a regular member whose hold queue output is managed by a super member, trace status is updated for both the regular member and the super member. Trace status is only updated on the IMS that processes the command. If the member specified is a regular member whose hold queue output is managed by a super member, and the trace status cannot be updated for both the regular member and the super member, the status is not updated for either member. The `DFS058I COMMAND COMPLETED EXCEPT` message is issued with the name of the regular member for which trace status could not be updated.

Examples

The following is an example of the /TRACE TMEMBER command:

Entry ET:

```
/TRACE SET ON TMEMBER CLIENT1 TPIPE TPIPESY
```

Response ET:

```
DFS058I 15:45:05 TRACE COMMAND COMPLETED   SYS3
DFS996I *IMS READY*  SYS3
```

/TRACE TRAN command

Use the /TRACE TRAN command to trace the DL/I portion of Data Communications (DC) for a specific transaction.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands" on page 615
- "Examples" on page 615

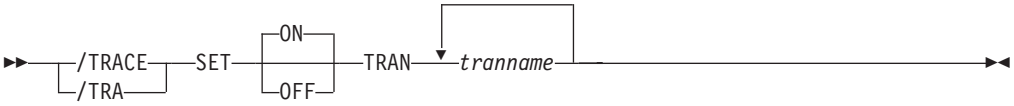
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 305. Valid environments for the /TRACE TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
SET	X	X	X
TRAN	X		X

Syntax



Usage notes

Each DL/I call to a TPPCB, issued by the user application program, is traced on entry to and exit from the DC call handler DFSDLA30. Upon entry to DFSDLA30, a type 6701-LA3A record is written; upon exit from DFSDLA30, a type 6701-LA3B record is written.

Additionally, in a shared-queues environment, a type 6701-ITQA record is written by module DFSITQS0 whenever a notification is received from CQS that there are messages available for the transaction.

Each of type 6701-LA3A and 6701-LA3B records will contain the following items, if applicable:

- TPPCB
- Up to 64 bytes of the I/O area
- Scheduler message block (SMB)
- Partition specification table (PST)

Each type 6701-ITQA record will contain the following items:

- IMSID of the IMS that wrote the record
- The AWE that was queued to DFSITQS0 to notify it that the transaction had messages to process
- SMB

Equivalent IMS type-2 commands

The following table shows variations of the /TRACE TRAN command and the IMS type-2 commands that perform similar functions.

Table 306. Type-2 equivalents for the /TRACE TRAN command

Task	/TRACE TRAN command	Similar IMS type-2 command
Starts the tracing of a transaction.	/TRACE SET ON TRAN *tranname*	UPDATE TRAN NAME(*tranname*) START(TRA)
Stops the tracing of a transaction.	/TRACE SET OFF TRAN *tranname*	UPDATE TRAN NAME(*tranname*) STOP(TRA)

Examples

The following is an example of the /TRACE TRAN command:

To log the PCB, I/O area, and PST whenever module DFSDLA30 is invoked to process transaction APPLE:

Entry ET:

```
/TRACE SET ON TRAN APPLE
```

Response ET:

```
DFS058I TRACE COMMAND COMPLETED
```

Related reference:

"UPDATE TRAN command" on page 795

/TRACE TRAP command

Use the /TRACE TRAP command to detect overwrites of MFS blocks. When /TRACE is used, IMS attempts to detect overwrites in the MFS blocks.

If an overwrite occurs, IMS sends a warning message.

Subsections:

- "Environment" on page 616
- "Syntax" on page 616

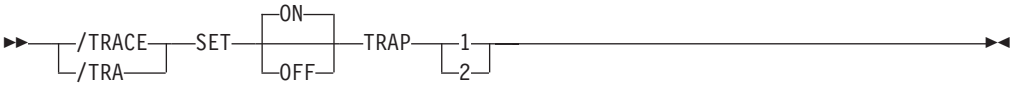
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 307. Valid environments for the /TRACE TRAP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
SET	X	X	X
TRAP	X		X

Syntax



Related reference:

Trace records (Diagnosis)

/TRACE UNITYPE command

Use the /TRACE UNITYPE command to trace events that are related to the physical terminals of the specified type.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 617
- "Usage notes" on page 618

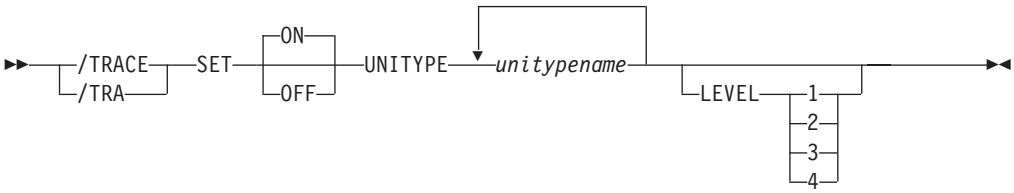
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 308. Valid environments for the /TRACE UNITYPE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
LEVEL	X		X
SET	X	X	X
UNITYPE	X		X

Syntax



Keywords

The following keywords are valid for the /TRACE UNITYPE command:

LEVEL

Expands the LINE, LINK, NODE, or UNITYPE trace functions. The LEVEL specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

LEVEL indicates the extent of the control block trace information desired. The indicated control blocks are only traced at relevant times. All levels are inclusive of numerically lower levels. The following list displays the levels and their associated blocks.

Level	Blocks
1	CLB (DECB) or LLB(MSC)
	CTB or LTB(MSC)
	IOB (for non-VTAM lines) or IOSB (MSC for channel-to-channel links)
2	CNT or LNB(MSC)
	CXB
	CRB
	CIB
	CCB
	PD stack
3	queue manager buffers
	Input/output line buffers
	LXB (for channel-to-channel links and processor storage to processor storage)
4	save area sets (IMS dispatching)

If the first /TRACE SET ON command does not specify LEVEL, a default of 4 is used. Specifying LEVEL on subsequent commands will change the defaults.

MODULE

Is used to expand the LINE, LINK, NODE, or UNITYPE trace functions. The MODULE specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

MODULE indicates which modules are to have their control blocks traced.

ALL Both device-dependent module (DDM) and MFS

DDM Communication analyzer and device-dependent module interfaces

MFS Communication analyzer and Message Format Service module interfaces

If the first /TRACE SET ON command does not specify MODULE, a default of ALL will be used. Specifying MODULE on subsequent commands will change the defaults.

Usage notes

The UNITYPE keyword is used to trace all terminals of a specific type. Parameters (*unitypename*) are similar to the identifiers displayed in the TYPE column by the /DISPLAY NODE and /DISPLAY LINE/PTERM commands. The following table shows the terminal types for UNITYPE parameters.

Table 309. UNITYPE parameters and terminal types

UNITYPE parameter	Terminal type
2260R	2260/2265 REMOTE
3286	3284/3286
2980	2980
3270R	3270 REMOTE
3270L	3270 LOCAL
RDR/PTR	LOCAL SYSIN/SYSOUT
FIN	3600
3277	3270 VTAM
SLU1	SLU TYPE 1
SLU2	SLU TYPE 2
SLUP	SLU TYPE P
LUT6	LU TYPE 6
NTO	NTO
CONSOLE	z/OS SYSTEM CONSOLE
TWX	TWX SWITCHED
3275SW	3270 SWITCHED
MSCBSC	Binary Synchronous Communication
MSCMTM	Memory to Memory Communication
MSCCTC	Channel to Channel Adapter
MSCVTAM	MSC VTAM

If global resource information is kept in Resource Manager, /TRACE UNITYPE sets a global trace status for all of the nodes of a specific type. This requires that the inactive static nodes be processed on every IMS system. The UNITYPE keyword is similar to specifying a generic parameter. If global resource information is not kept in Resource Manager, /TRACE UNITYPE sets the trace status locally.

Chapter 28. /UNLOCK commands

/UNLOCK commands release resources that, in most cases, have been previously locked by the /LOCK command.

If the terminals are on a switched communication network and a physical or logical terminal disconnection occurs, an implied /UNLOCK is processed against the physical terminal and inquiry logical terminal.

When using ISC, the /UNLOCK command can only be used with logical terminals assigned to allocated users.

This command can be issued to an IMSplex using the Batch SPOC utility.

/UNLOCK SYSTEM is only valid if it is entered from the master terminal or from the system console on an XRF system.

A resource name can be defined with password protection in SAF for the DATABASE, LTERM, PROGRAM, and TRANSACTION keywords. If the parameter, LOCKSEC=Y (N is the default) is specified on the DFSPBxxx IMS.PROCLIB member, IMS calls the SAF and user exit. If the resource is not defined to SAF, or is defined and is authorized to the user, the command is processed. If the resource is defined to SAF but not authorized for use, the command is rejected with a DFS3689W message.

The password associated with a signed on user and specified after a command resource parameter will be used to perform a reverification check, if the resource is defined to RACF with 'REVERIFY' specified in the APPLDATA field. Passwords can be mixed case or uppercase depending on what is specified on the PSWDC keyword in the DFSPBxxx IMS.PROCLIB member. If the resource passes the RACF authorization check, and RVFY=Y is specified as an IMS startup parameter, IMS will verify that the password following the parameter is the same as the password entered during signon for the user that entered the command. If 'REVERIFY' is specified for a resource, but a password is not provided, or the wrong password is provided, the command processing for that resource will be rejected.

Subsections:

- "/UNLOCK DB command" on page 620
- "/UNLOCK LTERM command" on page 621
- "/UNLOCK NODE command" on page 622
- "/UNLOCK PGM command" on page 623
- "/UNLOCK PTERM command" on page 624
- "/UNLOCK SYSTEM command" on page 625
- "/UNLOCK TRAN command" on page 627

© Copyright IBM Corp. 1974, 2015

/UNLOCK DB command

The /UNLOCK DB command specifies the database to be unlocked.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands" on page 621
- "Examples" on page 621

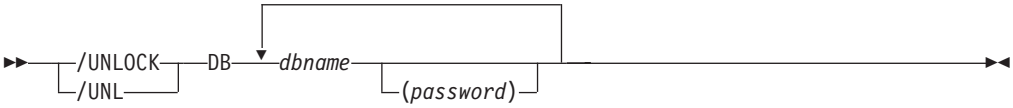
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 310. Valid environments for the /UNLOCK DB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
DB	X	X	

Syntax



Usage notes

The output of the /UNLOCK DB command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages: DFS0488I, DFS3466I, DFS132

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB with the exception of the /START DB and the UPDATE DB START(ACCESS) command. A HALDB partition is not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

The HALDB partition reflects conditions such as STOPPED, LOCKED, or NOTOPEN. When a HALDB partition is stopped, it must be explicitly started again. Commands with the keyword ALL and commands against a HALDB master do not change the STOPPED and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is /DBR on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. Only the HALDB master displays STOPPED (each HALDB partition does not display STOPPED

unless it was itself stopped). If a /DBR command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

Each partition inherits the access limitations of its HALDB master. If the /DBD command is issued against a HALDB master, all of its partitions close. A subsequent reference to any of the partitions results in the partition opening for input, although the partition's access might be UPDATE or EXCLUSIVE. The DBRC authorization state reflects the limited access.

This command can be issued by APPC and OTMA clients.

/UNLOCK DB is valid only if entered from the master terminal, the system console, a TCO script, or from an AOI application program.

While the database is being quiesced, this command cannot be processed successfully.

Equivalent IMS type-2 commands

The following table shows variations of the /UNLOCK DB command and the IMS type-2 commands that perform similar functions.

Table 311. Type-2 equivalents for the /UNLOCK DB command

Task	/UNLOCK DB command	Similar IMS type-2 command
Unlocks a database.	/UNLOCK DB *dbname*	UPDATE DB NAME(*dbname*) SET(LOCK(OFF))

Examples

The following is an example of the /UNLOCK DB command:

Entry ET:

```
/UNLOCK DB TREEFARM
```

Response ET:

```
DFS058I  UNLOCK COMMAND COMPLETED
```

Explanation: Database TREEFARM is unlocked and can be used.

Related reference:

"UPDATE DB command" on page 660

/UNLOCK LTERM command

The /UNLOCK LTERM command specifies the logical terminal to be unlocked. This keyword applies only to the entering physical terminal and to logical terminals assigned to that physical terminal.

Subsections:

- "Environment" on page 622
- "Syntax" on page 622
- "Usage notes" on page 622

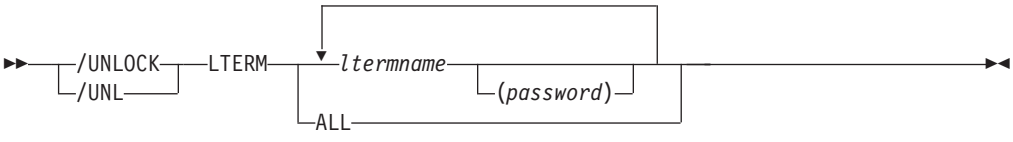
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 312. Valid environments for the /UNLOCK LTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
LTERM	X		X

Syntax



Usage notes

The /UNLOCK LTERM ALL command can only be used when all of the logical terminals associated with the entering physical terminal do not have passwords.

/UNLOCK LTERM applies only to the entering physical terminal.

/UNLOCK LTERM is not allowed from the OM API.

/UNLOCK NODE command

The /UNLOCK NODE command specifies the VTAM node to be unlocked. This keyword applies only to the entering physical terminal and to logical terminals assigned to that physical terminal.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes" on page 623

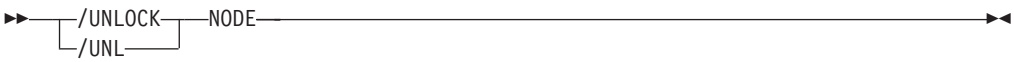
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 313. Valid environments for the /UNLOCK NODE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
NODE	X		X

Syntax



Usage notes

/UNLOCK NODE applies only to the entering physical terminal.

/UNLOCK NODE is not allowed from the OM API.

/UNLOCK PGM command

The /UNLOCK PGM command specifies the application program to be unlocked.

This command can be issued by APPC and OTMA clients.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands"
- "Examples" on page 624

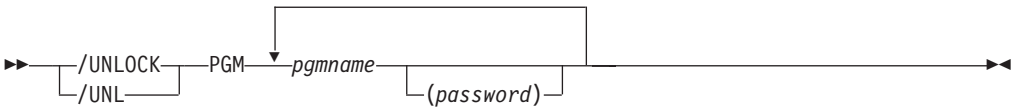
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 314. Valid environments for the /UNLOCK PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
PGM	X	X	X

Syntax



Usage notes

The /UNLOCK PGM command is valid only if entered from the master terminal, the system console, a TCO script, or from an AOI application program.

Equivalent IMS type-2 commands

The following table shows variations of the /UNLOCK PGM command and the IMS type-2 commands that perform similar functions.

Table 315. Type-2 equivalents for the /UNLOCK PGM command

Task	/UNLOCK PGM command	Similar IMS type-2 command
Unlocks a program.	/UNLOCK PGM *pgmname*	UPDATE PGM NAME(*pgmname*) SET(LOCK(OFF))

Examples

The following is an example of the /UNLOCK PGM command:

Entry ET:

```
/UNLOCK PGM APPLETRE
```

Response ET:

```
DFS058I  UNLOCK COMMAND COMPLETED
```

Explanation: Application program APPLETRE is unlocked and can be executed.

Related reference:

"UPDATE PGM command" on page 755

/UNLOCK PTERM command

The /UNLOCK PTERM command specifies the physical terminal to be unlocked. This keyword applies only to the entering physical terminal and to logical terminals assigned to that physical terminal.

This command can be issued by APPC and OTMA clients.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes" on page 625
- "Examples" on page 625

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 316. Valid environments for the /UNLOCK PTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
PTERM	X		X

Syntax

Usage notes

/UNLOCK PTERM applies only to the entering physical terminal.

/UNLOCK PTERM is not allowed from the OM API.

Examples

The following is an example of the /UNLOCK PTERM command:

Entry ET:

```
/UNLOCK PTERM
```

Response ET:

```
DFS058I  UNLOCK COMMAND COMPLETED
```

Explanation: The physical terminal from which the command is entered is unlocked.

/UNLOCK SYSTEM command

The /UNLOCK SYSTEM command notifies a newly created active system in an XRF complex that I/O prevention is complete.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 626

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 317. Valid environments for the /UNLOCK SYSTEM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
SYSTEM	X		X

Syntax

Usage notes

I/O prevention is initiated at takeover to ensure that the failing active system cannot write to the databases. The alternate system then initiates I/O toleration to ensure database integrity and to enable new transaction processing as soon as possible. /UNLOCK SYSTEM ends the I/O toleration phase of processing.

/UNLOCK SYSTEM is valid only if it is entered from the master terminal or from the system console on an XRF system.

Examples

The following is an example of the /UNLOCK SYSTEM command:

The following three figures illustrate the use of the /UNLOCK SYSTEM command on a newly created active system. Each figure is a formatted master screen for the newly created active system IMSB.

The following figure shows a screen of a newly created active system in the I/O toleration phase of processing (awaiting I/O prevention). Database DD41M803 has an I/O toleration EEQE.

```
02/05/15 16:19:03  RSENAME: DFSRSENM  ACTIVE  AWAITING I/O PREVENTION  IMSB
 DFS994I *CHKPT 85135/152931**SIMPLE**
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:   11
DFS3804I LAST CHKPT ID VALID FOR RESTART: 85135/161847-BUILDQ: 85135/161213
DFS994I TAKEOVER COMPLETED.
DFS3859I 16:18:29 PRIORITY 4 SESSIONS SWITCHED.
DFS3860I 16:18:29 ALL TERMINAL SESSIONS SWITCHED.
--------------------------------------------------------------------------------

     DATABASE
    DD41M803
        ERROR DD  TYPE    BLOCK
        DD41M803  IOT     0000003F
    *85135/161902*
                                                      PASSWORD:
/dis db dd41m803 bkerr
```

Figure 3. I/O toleration phase of processing

The following figure shows a screen of the use of the /UNLOCK SYSTEM command to notify the newly created active system that I/O prevention is complete (the XRF system status line now indicates that processing is no longer degraded by I/O toleration).

```
02/05/15 16:34:14  RSENAME: DFSRSENM  ACTIVE                          IMSB
 DFS994I *CHKPT 85135/161847**SIMPLE**
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:   11
DFS3804I LAST CHKPT ID VALID FOR RESTART: 85135/161847-BUILDQ: 85135/161213
 DFS058 16:34:14 UNLOCK COMMAND IN PROGRESS
DFS0488I - UNLOCK COMMAND COMPLETED.  RC = 00
DFS3860I 15:29:19 ALL TERMINAL SESSIONS SWITCHED.
--------------------------------------------------------------------------------


                                                      PASSWORD:
/unlock system
```

Figure 4. /UNLOCK SYSTEM command

The following figure is a screen that shows that the I/O toleration EEQE for database DD41M803 has been deleted as part of /UNLOCK SYSTEM processing.

```
02/05/15 16:35:00  RSENAME: DFSRSENM  ACTIVE                                IMSB
 DFS994I *CHKPT 85135/161847**SIMPLE**
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:   11
DFS3804I LAST CHKPT ID VALID FOR RESTART: 85135/161847  -  BUILDQ: 85135/161213
DFS058 16:34:14 UNLOCK COMMAND IN PROGRESS
DFS0488I - UNLOCK COMMAND COMPLETED.  RC = 00
DFS3860I 16:18:29 ALL TERMINAL SESSIONS SWITCHED.
--------------------------------------------------------------------------------

     DATABASE
    DD41M803
        NO EEQE OR INCOMPLETE BACKOUT INFORMATION AVAILABLE
    *85135/163500*
                                                    PASSWORD:
/dis db dd41m803 bkerr
```

Figure 5. EEQE deleted as part of /UNLOCK SYSTEM processing

Related concepts:

Takeover phase of the XRF process (System Administration)

/UNLOCK TRAN command

The /UNLOCK TRAN command specifies the transaction code to be unlocked.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands" on page 628
- "Examples" on page 628

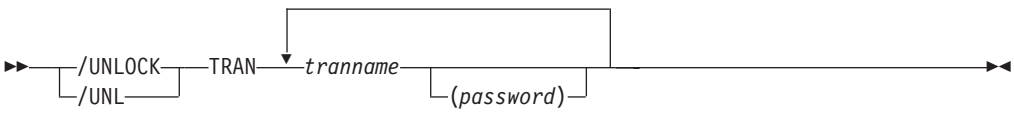
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 318. Valid environments for the /UNLOCK TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
TRAN	X		X

Syntax



Usage notes

The /UNLOCK TRAN command cannot be used for Fast Path exclusive or CPI Communications driven transaction programs.

This command can be issued by APPC and OTMA clients.

The /UNLOCK TRAN command is valid only if entered from the master terminal, the system console, a TCO script, or from an AOI application program.

Equivalent IMS type-2 commands

The following table shows variations of the /UNLOCK TRAN command and the IMS type-2 commands that perform similar functions.

Table 319. Type-2 equivalents for the /UNLOCK TRAN command

Task	/UNLOCK TRAN command	Similar IMS type-2 command
Unlocks a transaction.	/UNLOCK TRAN *tranname*	UPDATE TRAN NAME(*tranname*) SET(LOCK(OFF))

Examples

The following is an example of the /UNLOCK TRAN command:

Entry ET:

```
/UNLOCK TRAN SEED
```

Response ET:

```
DFS058I  UNLOCK COMMAND COMPLETED
```

Explanation: Transaction SEED is unlocked and can be scheduled.

Related reference:

"UPDATE TRAN command" on page 795

Chapter 29. UPDATE commands

Use the IMS UPDATE commands to update IMS resources or resource descriptors.

These commands can be issued through TSO SPOC or the Manage Resources options in the IMS Applications menu. These commands can also be issued to an IMSplex by using the Batch SPOC utility.

UPDATE commands are:

- "UPDATE AREA command"
- "UPDATE DATAGRP command" on page 643
- "UPDATE DB command" on page 660
- "UPDATE DBDESC command" on page 686
- "UPDATE IMS command" on page 692
- "UPDATE LE command" on page 701
- "UPDATE MSLINK command" on page 707
- "UPDATE MSNAME command" on page 717
- "UPDATE MSPLINK command" on page 724
- "UPDATE ODBM commands" on page 729
- "UPDATE OLREORG command" on page 745
- "UPDATE OTMADESC command" on page 750
- "UPDATE PGM command" on page 755
- "UPDATE PGMDESC command" on page 769
- "UPDATE POOL command" on page 779
- "UPDATE RTC command" on page 782
- "UPDATE RTCDESC command" on page 789
- "UPDATE TRAN command" on page 795
- "UPDATE TRANDESC command" on page 829

UPDATE AREA command

Use the UPDATE AREA command to change the status of area resources.

When the UPDATE AREA command is issued, the command is processed only by the IMS to which it is routed. The command does not preload or preopen areas on other IMS systems in the IMSplex that share the area. The UPDATE AREA command is routed by OM. OM routes the command to all active DB/DC or DBCTL IMS systems, unless specific routing is specified. OM selects one IMS as the command master.

Subsections:

- "Environment" on page 630
- "Syntax" on page 630
- "Keywords" on page 631
- "Usage notes" on page 634
- "Equivalent IMS type-1 commands" on page 635

© Copyright IBM Corp. 1974, 2015

- "Output fields" on page 635
- "Return, reason, and completion codes" on page 636
- "Examples" on page 640

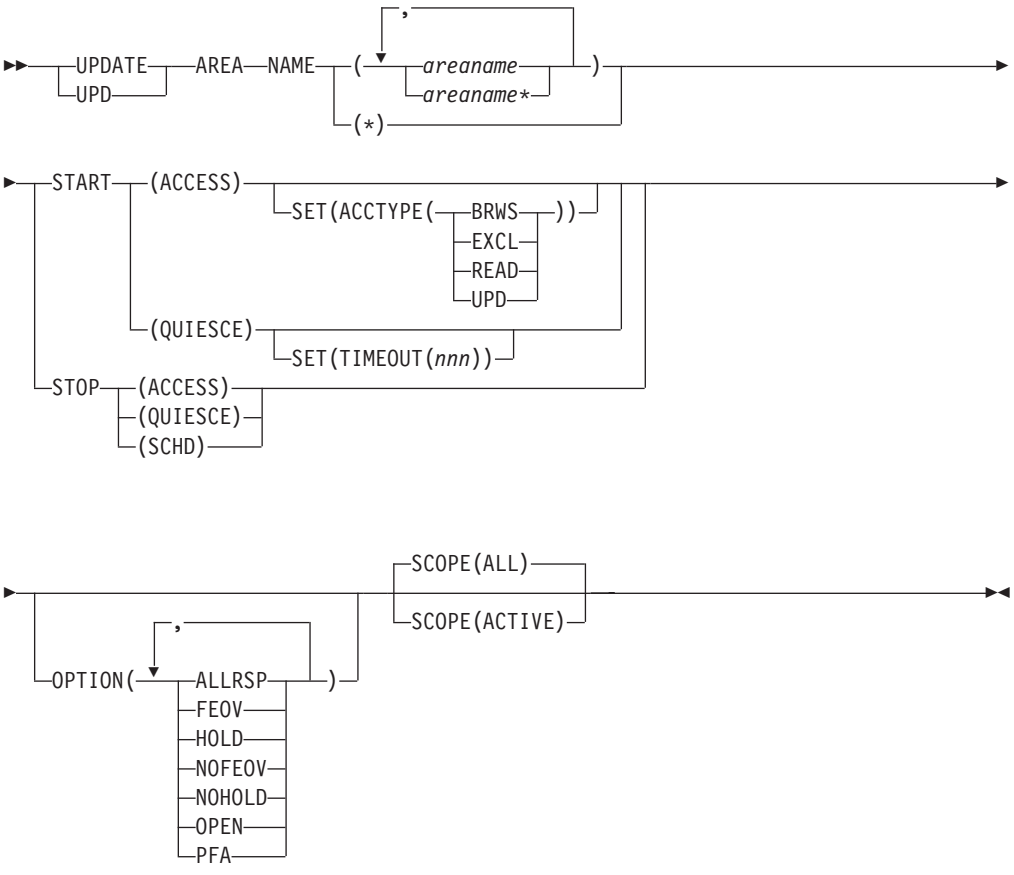
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE AREA command and keywords can be issued.

Table 320. Valid environments for the UPDATE AREA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE AREA	X	X	
NAME	X	X	
OPTION	X	X	
SCOPE	X	X	
START	X	X	
STOP	X	X	

Syntax



Keywords

The following keywords are valid for the UPDATE AREA command:

NAME()
Specifies the names of the specific areas that are to be processed or a group of areas to be processed.

Wildcard parameters can be specified. The area names that match the generic or wildcard parameter are processed. For specific or wildcard names, response lines are returned for all the area names that are processed.

NAME(*) indicates that the command is to be applied to all the areas in the system. When the NAME(*) is specified, the response lines are returned for only the area names that resulted in an error. If OPTION(ALLRSP) is specified with NAME(*), response lines are returned for all the area names that are processed.

OPTION()
Specifies the additional functions to be performed. Following is a list of additional functions:

ALLRSP
Indicates that the response lines are to be returned for all areas that are processed on the command. ALLRSP is only valid with NAME(*). The default action is to return response lines only for the areas that resulted in an error.

FEOV Indicates to force end of volume after the command processing is complete. The IMS log switches to the next OLDS, and a simple checkpoint is taken unless specified with STOP(QUIESCE). This switch is marked as a recovery point for log archiving purposes. A simple checkpoint is not taken when specified with STOP(QUIESCE).

FEOV is valid only with START(QUIESCE) OPTION(NOHOLD), STOP(ACCESS), or STOP(QUIESCE), and is not valid on an RSR tracker. OPTION(FEOV) is the default when specified with START(QUIESCE) OPTION(NOHOLD) or STOP(QUIESCE).

For quiesce processing, the end of volume will be forced after the quiesce point has been reached, causing the logs to be switched before the quiesce is released, so that any new updates occur on the new IMS log. This occurs during the stop quiesce processing of a START(QUIESCE) OPTION(NOHOLD) or when a STOP(QUIESCE) is issued to release a quiesce from a previous START(QUIESCE) OPTION(HOLD).

HOLD
Specifies that after the quiesce has been achieved successfully the area should remain quiesced. A subsequent STOP(QUIESCE) would be required to release the quiesce on the DEDB area. HOLD is valid only with the START(QUIESCE) keyword. This keyword is mutually exclusive with the NOHOLD keyword.

NOFEOV
Indicates to not force end of volume after the command processing is complete. The IMS log does not switch to the next OLDS and a simple checkpoint is not taken.

NOFEOV is valid only with START(QUIESCE) OPTION(NOHOLD), STOP(ACCESS), or STOP(QUIESCE). OPTION(NOFEOV) is the default when specified with STOP(ACCESS).

NOHOLD
Specifies that after the quiesce has been achieved successfully the area should automatically release the quiesce. If neither HOLD nor NOHOLD is specified, NOHOLD is assumed. NOHOLD is valid only with the START(QUIESCE) keyword. This keyword is mutually exclusive with the HOLD keyword.

OPEN Specifies to open the DEDB area data sets that are specified on the NAME() parameter, even if the area is not registered on DBRC as PREOPEN. The OPTION(OPEN) is processed locally by all IMS systems that receive the command and is not maintained as a global status in RM.

PFA Sets or resets the prevent further authorization (PFA) status in the RECON data set for the DEDB area. Use the PFA option with UPDATE AREA START(ACCESS) to enable access to a DEDB area. Use the PFA option with UPDATE AREA STOP (ACCESS|SCHD) to prevent access to a DEDB area. If you specify OPTION(PFA), the command master IMS updates the RECON only if the command is successful at the master IMS.

SCOPE()
Specifies where IMS should apply the change. The default is ALL.

ALL
Changes are applied to the active IMS systems to which the command is routed. If you specify that global area status is to be maintained, changes are also applied globally by updating the value maintained by RM. The RM status is updated only by the command master IMS. If global area status is not maintained, the command action is same as the SCOPE(ACTIVE) command.

This option does not apply to the quiesce function. The scope of a quiesce is always all instances of the area usage across the IMSplex. There is not a global status of QUIESCE for a DEDB area. Using SCOPE(ALL) for a START(QUIESCE) or STOP(QUIESCE) does not change the global status for the area.

ACTIVE
Changes are applied to the active IMS systems to which the command is routed to. Any global status information maintained in the RM resource structure is not changed by the SCOPE(ACTIVE) command.

This option does not apply to the quiesce function. The scope of a quiesce is always all instances of the area usage across the IMSplex. There is no quiesce which would only apply to a subset of the IMSplex.

The UPDATE AREA command keywords that update information locally in the active IMS system and globally in RM when SCOPE(ALL) is specified include:

- START(ACCESS)
- STOP(SCHD)
- STOP(ACCESS)

SET() Specifies the attribute values to be changed.

ACCTYPE()
Specifies the access intent for the named area. This keyword can be specified only if START(ACCESS) is specified. This keyword overrides the database access intent of its DEDB.

You can specify one of the following keywords:

BRWS
Specifies that the named area is available for read-only processing on this IMS subsystem. The only programs that can use the area on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data by using the GO processing option might access uncommitted data because a sharing IMS subsystem could be updating the area. The area is opened for read-only processing.

EXCL
Specifies that the named area is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the area is registered to DBRC.

READ
Specifies that the named area is available for read processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the area. ACCTYPE of READ differs from ACCTYPE of BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The area is opened for read.

UPD
Specifies that the named area is available for update as well as read processing in the IMS subsystem.

TIMEOUT(*nnn*)
Specifies the number of seconds to wait before a timeout occurs in a DEDB area quiesce. The timeout value can be 1 - 999 seconds. The TIMEOUT parameter value can override the DBQUIESCETO parameter in the DFSCGxxx member of the IMS PROCLIB data set. If the TIMEOUT parameter is omitted and the DBQUIESCETO parameter is not specified, the default timeout value is 30 seconds. The TIMEOUT keyword is valid only with the START(QUIESCE) keyword.

START()
Specifies the attributes that are to be started.

ACCESS
Specifies the specific areas of a DEDB to be allocated or reallocated.

An UPDATE AREA START(ACCESS) can be issued on an RSR tracker to resume tracking for those areas that were stopped or had tracking errors. In addition, Online Forward Recovery (OFR) is started for areas that are not current with mainline tracking.

For Virtual Storage Option (VSO) areas, the UPDATE AREA START(ACCESS) can be used to preopen areas that are defined with the PREOPEN option. For VSO areas defined with the PRELOAD option, the command causes the areas to be loaded into

the z/OS data space or an XES structure depending on the share level of the area. The command has no effect on VSO areas that are in virtual storage.

QUIESCE
Specifies that the DEDB areas named on the command are to be quiesced to establish a new recovery point. The scope of a quiesce is always all instances of the area usage across the IMSplex. There is no quiesce that would apply only to a subset of the IMSplex.

STOP()
Specifies the attributes that are to be stopped.

ACCESS
Stops the access and updating of the specified DEDB areas and closes them.

The UPDATE AREA STOP(ACCESS) command for VSO areas removes the areas from the data space or XES structure and forces updates to be written back to DASD.

An UPDATE AREA START(ACCESS) command is required to open and reallocate the areas closed by the UPDATE AREA STOP(ACCESS) command.

QUIESCE
Specifies that the DEDB areas named on the command should be made available again by releasing the quiesce on the areas.

Unlike START(QUIESCE), where each area resource listed must be quiesced in order for the command to complete successfully, STOP(QUIESCE) continues to process each listed resource even if some resources are not in quiesced state or cannot be released from quiesced state. For those resources that are not in quiesced state or cannot be released from quiesced state, the command returns a response line for each of those resources.

SCHD Specifies that the data sets associated to the areas are to be closed and deallocated.

Note: An UPDATE AREA START(ACCESS) is required to reallocate the areas stopped by the UPDATE AREA STOP(SCHD) command.

If UPDATE AREA STOP(SCHD) is processed during HSSP processing, the area will be released after the current commit processing completes. Any image copy option in process when the command is issued can affect the continued system operation.

For VSO DEDB areas that are in a z/OS data space or XES structure, the UPDATE AREA STOP(SCHD) command action results in the removal of the VSO areas from the data space or XES structure and the writing of updates to DASD.

Usage notes

The UPDATE AREA command can only be specified using the OM API. The command is also not allowed on the XRF alternate.

The UPDATE AREA STOP(SCHD) command and the UPDATE AREA START(ACCESS) SET(ACCTYPE()) command are not allowed on any RSR tracker.

However, the commands UPDATE AREA START(ACCESS) or UPDATE AREA STOP(ACCESS) are allowed only on a database level RSR tracker. The UPDATE AREA START(ACCESS) and UPDATE AREA STOP(SCHD) commands are recoverable, and a X'22' log record is written but the UPDATE AREA STOP(ACCESS) command is not recoverable.

A new log record, X'594C', is written for every DEDB area to which the UPDATE AREA command applies. It also includes global status and the global command time stamp.

The UPDATE AREA command is enhanced to return CCTXT with a non-zero completion code. The CCTXT can be up to 32 bytes, and it includes information about what the completion code means. The UPDATE AREA SCOPE(ALL) command returns a response line with the completion code for the global status update. The CCTXT for that code is GBL CC.

While the database is being quiesced, this command with keywords other than STOP(QUIESCE) cannot be processed successfully.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE AREA command and the type-1 IMS commands that perform similar functions.

Table 321. Type-1 equivalents for the UPDATE AREA command

UPDATE AREA command	Similar IMS type-1 commands
`UPDATE AREA NAME(name) STOP(ACCESS)`	`/DBR AREA areaname`
`UPDATE AREA NAME(name) START(ACCESS)`	`/START AREA areaname`
`UPDATE AREA NAME(name) STOP(SCHD)`	`/STOP AREA areaname`

Output fields

The following table shows the UPDATE AREA output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. ERR appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 322. Output fields for the UPDATE AREA command

Short label	Keyword	Meaning
AREA	N/A	Area name. The area name is always returned.
CCTXT	*error*	The completion code text that briefly explains the meaning of the completion code.
CC	N/A	Completion code for the line of output. Completion code is always returned.

Table 322. Output fields for the UPDATE AREA command (continued)

Short label	Keyword	Meaning
GBL	SCOPE(ALL)	Indicates that the response line is for the global update.
EERT	ERR	Error text returned to add more meaning to the completion code and may include a return code from a service. The error text may be returned for a non-zero completion code.
MBR	N/A	The IMSplex member that built the output line. The IMS identifier of the IMS for which the area information is displayed. The IMS identifier is always returned.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the UPDATE AREA command. The OM return and reason codes that may be returned as a result of the UPDATE AREA command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 323. Return and reason codes for the UPDATE AREA command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE AREA command completed successfully.
X'00000008'	X'00002014'	The UPDATE AREA command is not processed because an invalid character is found in the area name parameter.
X'00000008'	X'00002040'	Either more than one keyword or an invalid combination of filters was specified on the UPDATE AREA command. For example, the following keyword combinations are incorrect: • START(ACCESS) and STOP(ACCESS) • STOP(ACCESS) and OPTION(OPEN) • STOP(SCHD) and OPTION(OPEN) • OPTION(NOFEOV,FEOV) Check the input command and reenter the correct combinations.
X'00000008'	X'00005000'	The UPD AREA command processing terminated because IMODULE GETSTOR storage could not be obtained.
X'0000000C'	X'00003000'	The UPDATE AREA command is successful for at least one resource name. The UPDATE AREA command was not successful for one or more resource names. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the UPDATE AREA command are listed in Table 324 on page 637.

Table 323. Return and reason codes for the UPDATE AREA command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003004'	The UPDATE AREA command was not successful for all the resource name(s) specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the UPDATE AREA command are listed in Table 324.
X'00000010'	X'0000400C'	The UPDATE AREA command is not processed because the command variation entered is not valid on the XRF alternate.
X'00000010'	X'00004014'	The UPDATE AREA command is not processed because the function is not valid on the RSR tracker.
X'00000010'	X'00004024'	The UPDATE AREA command is not processed because Fast Path is not installed.
X'00000010'	X'00004025'	The UPDATE AREA command is rejected because no Fast Path areas are defined.
X'00000010'	X'00004200'	Commands are not processed because IMS shutdown is in progress.
X'00000010'	X'00004400'	MINVERS in the RECON data sets is not 11.1.
X'00000014'	X'00005001'	The UPDATE AREA command processing terminated because WKAP or MAIN pool storage could not be obtained.
X'00000014'	X'00005004'	The UPDATE AREA command processing terminated because a DFSCOMD response buffer could not be obtained.
X'00000014'	X'00005008'	The UPDATE AREA command processing terminated because a DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	The UPDATE AREA command processing terminated because AWE could not be obtained.
X'00000014'	X'00005FFF'	The UPDATE AREA command processing terminated because of an internal error.
X'02000008'	X'0000203C'	An invalid TIMEOUT value is specified. The value must be a numeric value between 1 and 999.

The following table includes an explanation of the completion codes. Errors unique to the processing of UPDATE AREA command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 324. Completion codes for the UPDATE AREA command

Completion code	Completion code text	Meaning
0		The command completed successfully for the resource.
0C	COMMAND COMPLETE FOR NONE	This error is returned when all the areas could not be quiesced.
10	NO RESOURCE FOUND	No resource found.

Table 324. Completion codes for the UPDATE AREA command (continued)

Completion code	Completion code text	Meaning
11	DUPLICATE RESOURCE NAME	The resource name is specified multiple times on the command and is ignored.
17	ANOTHER CMD IN PROGRESS	This error is returned when the quiesce could not be started because another database command was in progress.
53	NO RM ADDRESS SPACE	This error is returned when the quiesce could not be started because the RM address space is not present.
55	NO FAST PATH INSTALLED	The command failed because Fast Path is not installed.
81	DBRC ERROR	This error is returned when an unexpected DBRC error occurs during the quiesce command.
90	INTERNAL ERROR	The command entered is not processed because of an internal error.
91	TIMEOUT ERROR	This error is returned when the quiesce could not be completed within the timeout period. The quiesce might be processing longer than the timeout value, or an IMSplex component involved in the process might have failed or hung. Check the state of the IMSplex components by, for example, issuing a QUERY IMSPLEX command or checking the z/OS system log.
92	COMMAND PROCESSING ERROR	The command entered is not processed because of a command error. A unique completion code could not be generated to explain the error. The message number and the return code that could not be converted to a completion code are listed in the error text.
A5	PREVENT FURTHER AUTH ON	The command entered is not processed, because the database or area is defined to DBRC as 'prevent further authorization'.
A9	DB OR AREA AUTHORIZATION ERROR	Area authorization to DBRC failed.
D1	DATABASE WRITE ERROR	This error is returned when the quiesce could not be completed because the database has a write error.
D2	DATABASE NEEDS BACKOUT	This error is returned when the quiesce could not be completed because the database needs backout.
D3	DATABASE OR AREA NEEDS RECOVERY	This error is returned when the quiesce could not be completed because the database needs recovery.
D9	COMMAND PROCESSING ERROR	The UPDATE AREA START(ACCESS) command could not be processed for the area name because the area open failed.
E0	DATABASE OR AREA IN RECOVERY	The command is not processed because the database or area is in recovery.
E5	PARTICIPANTS UNABLE TO QUIESCE	This error is returned when the quiesce could not be completed successfully across the IMSplex. The IMS with this completion code was the quiesce participant that was not able to be quiesced.

Table 324. Completion codes for the UPDATE AREA command (continued)

Completion code	Completion code text	Meaning
E6	QUIESCE COMMUNICATION FAILURE	This error is returned when the quiesce could not be completed because of a failure to communicate across the IMSplex. There could be a problem with RM, OM, or SCI that has caused the communication failure to occur.
E7	CMD NOT ALLOWED	This error is returned when the database command could not be processed because a quiesce command was in progress.
E8	DATABASE HAS INTENT TO REORGANIZE	This error is returned when the quiesce could not be started because the RECON data sets indicate that there is an intent to reorganize the database.
E9	DB IN WRONG STATE TO BE QUIESCED	This error is returned when the named resource is in the wrong state for quiesce processing to proceed.
EF	DATABASE IS IN ERROR	This error is returned when the quiesce could not be completed because the database is in error.
F0	NO AREA LOCK	The command processing failed because the area lock could not be obtained.
F1	AREA NOT STOPPED	The command entered is not processed because the area is not stopped.
F2	PRELOAD IS ACTIVE FOR AREA	The command is not processed because preload is active for the area.
F3	UNRESOLVED INDOUBTS FOR AREA	The command entered is not processed because unresolved indoubts exist for the area.
F4	ALLOCATION FAILED	Allocation failed for the area name.
F5	AREA NEEDS RECOVERY	The command processing failed because the area needs recovery.
F6	ADS NUMBER DISCREPANCY	The command failed for the AREA because there is a discrepancy between the number of ADS allocated by IMS and the number of ADS known to DBRC. Correct the discrepancy and reissue the command to deallocate the data sets.
F7	AREA IS NOT LOADED INTO CF	The area open failed and is not loaded into the Coupling Facility.
F8	AREA HAS I/O TOLERATED CI	The command is not processed because the area has an I/O tolerated CI.
F9	AREA HAS SECOND CI EEQE	The command is not processed because the area has a second CI EEQE.
FC	UTILITY ACTIVE ON AREA	This error is returned when the area is in use by a utility.
FD	AREA HAS EEQE	This error is returned when the area has an extended error queue element (EEQE).
FE	AREA HAS EQE	This error is returned when the area has an error queue element (EQE).
190	DB ACCESS LESS THAN AREA ACCESS.	The command fails for the resource, because the DEDB access is less than the area access.
191	AREA ACCESS ALREADY AT LEVEL.	The command fails for the resource, because the area access is already at the area access level.

Table 324. Completion codes for the UPDATE AREA command (continued)

Completion code	Completion code text	Meaning
192	AREA IN USE-UTIL.	The command fails for the resource, because a utility is running against the area.

Examples

The following are examples of the UPDATE AREA command:

Example 1 for UPDATE AREA command

TSO SPOC input:

```
UPD AREA NAME(DB21AR1*) STOP(ACCESS)
```

TSO SPOC output:

```
AreaName MbrName    CC
DB21AR1  IMS2        0
DB21AR1  SYS3        0
DB21AR10 IMS2        0
DB21AR10 SYS3        0
DB21AR11 IMS2        0
DB21AR11 SYS3        0
```

OM API input:

```
CMD(UPD AREA NAME(DB21AR1*) STOP(ACCESS))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.132 16:17:33.260435</statime>
<stotime>2003.132 16:17:33.281165</stotime>
<staseq>B968A333F5A93283</staseq>
<stoseq>B968A333FAB8DC83</stoseq>
<rqsttkn1>USRT005 10091733</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>AREA            </kwd>
<input>UPD AREA NAME(DB21AR1*) STOP(ACCESS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>AREA(DB21AR1 ) MBR(IMS2    ) CC(   0) </rsp>
```

```
<rsp>AREA(DB21AR10) MBR(IMS2    ) CC(   0) </rsp>
<rsp>AREA(DB21AR11) MBR(IMS2    ) CC(   0) </rsp>
<rsp>AREA(DB21AR1 ) MBR(SYS3    ) CC(   0) </rsp>
<rsp>AREA(DB21AR10) MBR(SYS3    ) CC(   0) </rsp>
<rsp>AREA(DB21AR11) MBR(SYS3    ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The command stops access to all the areas that match the area name specified and makes them unavailable. The areas data sets are closed and deallocated.

Example 2 of UPDATE AREA command

TSO SPOC input:

```
UPD AREA NAME(*) STOP(SCHD)
```

TSO SPOC output:

```
Log for . . : UPD AREA NAME(*) STOP(SCHD)
 IMSplex . . . . . : PLEX1
 Routing . . . . . :
 Start time. . . . : 2003.132 09:22:15.79
 Stop time . . . . : 2003.132 09:22:17.53
 Return code . . . : 00000000
 Reason code . . . : 00000000
 Command master. .  IMS2
```

OM API input:

```
CMD(UPD AREA NAME(*) STOP(SCHD))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.132 16:22:15.799896</statime>
<stotime>2003.132 16:22:17.536107</stotime>
<staseq>B968A44169058C8E</staseq>
<stoseq>B968A44310E6BE2D</stoseq>
<rqsttkn1>USRT005 10092215</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>AREA            </kwd>
<input>UPD AREA NAME(*) STOP(SCHD) </input>
</cmd>
</imsout>
```

Explanation: The command is routed to IMS2 and SYS3, and the command is successful at both IMS systems. No response lines are returned because the default action for NAME(*) is to return response lines only for the area names that resulted in an error. OPTION(ALLRSP) can be specified to obtain all the area names processed on the command.

Example 3 of UPDATE AREA command

TSO SPOC input:

```
UPD AREA NAME(DD01AR0) STOP(ACCESS) SCOPE(ALL)
```

TSO SPOC output:

```
AreaName  MbrName  CC  Global
DD01AR0   IMS1      0       Y
DD01AR0   IMS1      0
DD01AR0   IMS2      0
```

Explanation: The UPDATE AREA STOP(ACCESS) command for area DD01AR0 is successfully processed at IMS1 and IMS2. Command master IMS1 successfully updates the global status.

Example 4 of UPDATE AREA command

TSO SPOC input:

```
UPDATE AREA NAME(DB22AR0,DB22AR1,DB22AR2,DB22AR3) START(ACCESS) OPTION(OPEN)
```

TSO SPOC output:

```
AreaName MbrName    CC   CCText
DB22AR0  IMS1        0
DB22AR1  IMS1        0
DB22AR2  IMS1       A9   DB OR AREA AUTHORIZATION ERROR
DB22AR3  IMS1       A9   DB OR AREA AUTHORIZATION ERROR
```

Explanation: This command opens the DEDB area data sets that are specified on the NAME() parameter, even if the area is not registered on DBRC as PREOPEN. The OPTION(OPEN) is processed locally by all IMS systems that receive the command and is not maintained as a global status in RM.

Example 5 of UPDATE AREA command

TSO SPOC input:

```
UPDATE AREA NAME(AXYZ01) START(QUIESCE) OPTION(HOLD)
```

TSO SPOC output:

```
AreaName   MbrName    CC
AXYZ01     IM02        0
AXYZ01     IM01        0
AXYZ01     IM03        0
```

Explanation: This example is of a successful quiesce and hold for a DEDB area.

Example 6 of UPDATE AREA command

TSO SPOC input:

```
UPD AREA NAME(AXYZ01) STOP(QUIESCE)
```

TSO SPOC output:

```
AreaName   MbrName      CC
AXYZ01      IM03         0
AXYZ01      IM02         0
AXYZ01      IM01         0
```

Explanation: example is of releasing the quiesce for a DEDB area.

Example 7 of UPDATE AREA command

TSO SPOC input:

```
UPDATE AREA NAME(D0010001) START(ACCESS) SET(ACCTYPE(READ))
```

TSO SPOC output:

```
 AreaName MbrName    CC
 D0010001 IMS1        0
```

Explanation: DEDB DEDBJ0001 has database access of update. Area D0010001 of DEDB DEDBJ001 access is changed from update to read.

Related concepts:

Maintaining global information for databases, DEDB areas, and transactions (System Administration)

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

"/START AREA command" on page 456

"/STOP AREA command" on page 509

UPDATE DATAGRP command

Use the UPDATE DATAGRP command to allow changes to the status of the members of a data group.

Subsections:

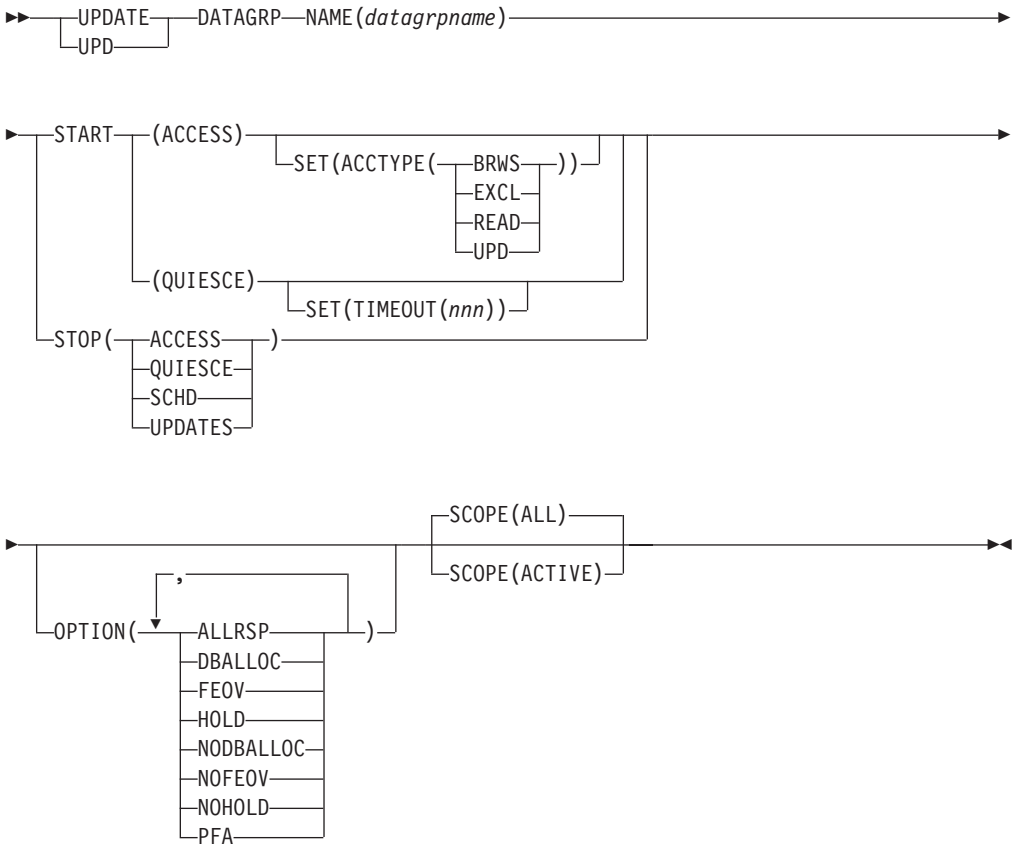
- "Environment"
- "Syntax" on page 644
- "Keywords" on page 644
- "Usage notes" on page 647
- "Equivalent IMS type-1 commands" on page 648
- "Output fields" on page 648
- "Return, reason, and completion codes" on page 649
- "Examples" on page 656

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE DATAGRP command and keywords can be issued.

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE DATAGRP	X	X	
NAME	X	X	
OPTION	X	X	
SCOPE	X	X	
START	X	X	
STOP	X	X	

Syntax



Keywords

The following keywords are valid for the UPDATE DATAGRP command:

NAME()
Specifies the name of the data group to be processed.

OPTION()
Specifies the additional functions to be performed. Following is a list of additional functions:

ALLRSP
Indicates that the response lines are to be returned for all the members of the data group (all databases, areas, or both) that are processed on the command.

The default action is to return response lines only for the databases and areas or both that resulted in an error. A response line with the data group name is always returned.

DBALLOC
Indicates that the databases in the data group are to be allocated when they are started.

DBALLOC can only be specified with START(ACCESS).

Only one of DBALLOC or NODBALLOC can be specified.

FEOV Indicates to force end of volume after the command processing is

complete. The IMS log switches to the next OLDS, and a simple checkpoint is taken unless specified with STOP(QUIESCE). This switch is marked as a recovery point for log archiving purposes. A simple checkpoint is not taken when specified with STOP(QUIESCE).

FEOV is valid only with START(QUIESCE) OPTION(NOHOLD) or STOP(ACCESS) or STOP(QUIESCE). FEOV is not valid on an RSR tracker. OPTION(FEOV) is the default when specified with START(QUIESCE) OPTION(NOHOLD) or STOP(QUIESCE).

For quiesce processing, the end of volume will be forced after the quiesce point has been reached, causing the logs to be switched before the quiesce is released, so that any new updates occur on the new IMS log. This occurs during the stop quiesce processing of a START(QUIESCE) OPTION(NOHOLD) or when a STOP(QUIESCE) is issued to release a quiesce from a previous START(QUIESCE) OPTION(HOLD).

HOLD
Specifies that after the quiesce has been achieved successfully the data group members should remain quiesced. A subsequent STOP(QUIESCE) would be required to release the quiesce on the data group members. This keyword is only valid with the START(QUIESCE) keyword. HOLD is mutually exclusive with NOHOLD.

NODBALLOC
Indicates that the databases in the data group are not to be allocated when they are started. The databases will be allocated when they are scheduled.

NODBALLOC can only be specified with START(ACCESS). NODBALLOC is the default action for UPDATE DATAGRP command if OPTION(DBALLOC) is not specified.

Only one of DBALLOC or NODBALLOC can be specified.

NOFEOV
Indicates to not force end of volume after the command processing is complete. The IMS log does not switch to the next OLDS, and a simple checkpoint is not taken.

NOFEOV is valid only with START(QUIESCE) OPTION(NOHOLD), STOP(ACCESS), or STOP(QUIESCE). OPTION(NOFEOV) is the default when specified with STOP(ACCESS).

NOHOLD
Specifies that after the quiesce has been achieved successfully the data group members should automatically release the quiesce. If you do not specify either HOLD or NOHOLD, NOHOLD is assumed. This keyword is only valid with the START(QUIESCE) keyword. NOHOLD is mutually exclusive with HOLD.

SCOPE()
Specifies where IMS should apply the change.

ACTIVE
SCOPE(ACTIVE) specifies that the changes are to be applied to the IMS systems that are currently active and to which the command is routed.

This option does not apply to the quiesce function. The scope of a quiesce is always all instances of the data group usage across the IMSplex. There is no quiesce which would only apply to a subset of the IMSplex.

ALL SCOPE(ALL) specifies that the changes are to be applied to the active IMS systems.

This option does not apply to the quiesce function. The scope of a quiesce is always all instances of the data group members usage across the IMSplex. There is not a global status of QUIESCE for a data group or its members. Using SCOPE(ALL) for a START(QUIESCE) or STOP(QUIESCE) does not change the global status for the data group members.

Currently, ACTIVE and ALL produce the same results.

SET() Specifies the attribute values to be changed or sets the database state.

ACCTYPE
Specifies the access intent for the named database. This keyword can be specified only if START(ACCESS) is specified. This keyword overrides the database access intent specified at system definition.

BRWS Specifies that the named database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data because another program could be updating the database. The database is opened for read-only processing.

EXCL Specifies that the named database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC.

READ Specifies that the named database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. ACCTYPE of READ differs from ACCTYPE of BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UPD Specifies that the named database is available for update as well as read processing in this IMS subsystem.

TIMEOUT(*nnn*)
Specifies the number of seconds to wait before a timeout occurs in a data group quiesce. The timeout value can be 1 - 999 seconds. The TIMEOUT parameter value can override the DBQUIESCETO parameter in the DFSCGxxx member of the IMS PROCLIB data set. If the TIMEOUT parameter is omitted and the DBQUIESCETO

parameter is not specified, the default timeout value is 30 seconds. The TIMEOUT keyword is valid only with the START(QUIESCE) keyword.

START()
Specifies the attributes that are to be started.

ACCESS
Specifies that the members of the specified data group name are to be started. See UPDATE AREA START(ACCESS) and UPDATE DB START(ACCESS) for a description of the actions performed on the databases and areas in the data group.

Additional functions to be performed along with START(ACCESS) can be specified using the OPTION keyword.

An UPDATE DATAGRP START(ACCESS) command with SET(ACCTYPE) or OPTION(DBALLOC | NODBALLOC | NOBACKOUT | OPEN | NOOPEN) is invalid on the RSR tracker.

QUIESCE
Specifies that the data group named on the command is to be quiesced to establish a new recovery point. The scope of a quiesce is always all instances of the data group usage across the IMSplex.

STOP()
Specifies the attributes to be stopped.

ACCESS
Specifies that offline processing is to be done for the members of the specified data group. See UPDATE AREA STOP(ACCESS) and UPDATE DB STOP(ACCESS) for a description of the actions performed on the databases and areas in the data group.

QUIESCE
Specifies that the data group named on the command should be made available again by releasing the quiesce on the data group members.

Unlike START(QUIESCE), where each data group resource listed must be quiesced in order for the command to complete successfully, STOP(QUIESCE) continues to process each listed resource even if some resources are not in quiesced state or cannot be released from quiesced state. For those resources that are not in quiesced state or cannot be released from quiesced state, the command returns a response line for each of those resources.

SCHD Specifies that the members of the specified data group name are to be stopped. See UPDATE AREA STOP(ACCESS) and UPDATE DB STOP(ACCESS) for a description of the actions performed on the databases and areas in the data group.

Usage notes

The UPDATE DATAGRP command can only be specified through the OM API and can be processed only by the DB/DC and DBCTL environments. In addition, the UPDATE DATAGRP command is not allowed on the XRF alternate.

When the UPDATE DATAGRP command is issued, it only applies to the IMS system to which it is routed. It does not apply to the other IMS systems in the IMSplex that share the database or areas in the data group. The routing of the

UPDATE DATAGRP command is done by OM. OM routes the command to all active DB/DC or DBCTL IMS systems, unless specific routing is specified. OM selects one IMS as the command master.

A response line is returned for the data group name from each IMS. Response lines are also returned for each data group member that resulted in an error. No response lines are returned for the data group members that are processed successfully unless OPTION(ALLRSP) is specified.

A *data group* is defined in the RECON data set by using the INIT.DBDSGRP command with the parameters GRPNAME and DBGRP (to define a DB group), MEMBERS (to define a DBDS group), or RECOVGRP (to define a recovery group). The DATAGRP keyword on the UPDATE command can specify either a DBDS group or a DB group (DL/I databases or DEDB areas).

If the ACCESS keyword is specified on the UPDATE DATAGRP command along with the DBDS group name, the ACCESS keyword is not applied to the Fast Path DEDB databases associated with the Fast Path DEDB areas in the DBDS group. If the intent is to use the ACCESS keyword for Fast Path DEDB databases, the DATAGRP parameter must be a data group name that does not contain area names.

While the database is being quiesced, this command with keywords other than STOP(QUIESCE) cannot be processed successfully.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE DATAGRP command and the type-1 IMS commands that perform similar functions.

Table 325. Type-1 equivalents for the UPDATE DATAGRP command

UPDATE DATAGRP command	Similar IMS type-1 command
`UPDATE DATAGRP NAME(name) STOP(ACCESS)`	`/DBR DATAGRP datagrpname`
`UPDATE DATAGRP NAME(name) START(ACCESS)`	`/START DATAGRP datagrpname`
`UPDATE DATAGRP NAME(name) STOP(SCHD)`	`/STOP DATAGRP datagrpname`

Output fields

The following table shows the output fields for the UPDATE DATAGRP. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. ERR appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 326. Output fields for UPDATE DATAGRP command

Short label	Keyword	Meaning
AREA	N/A	Area name. The Area name is returned if there are one or more areas in the data group.
CC	N/A	Completion code for the line of output. Completion code is always returned.
CCTXT	*error*	The completion code text that briefly explains the meaning of the completion code.
DB	N/A	Database name. The database name is returned if there are one or more databases in the data group.
DG	N/A	Data group name. The data group name is always returned.
ERRT	ERR	Error text returned to add more meaning to the completion code and may include a return code from a service. The error text is only returned if the completion code is non-zero.
MBR	N/A	The IMSplex member that built output line. The IMS identifier of the IMS for which the database information is displayed. The IMS identifier is always returned.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the UPDATE DATAGRP command. The OM return and reason codes that may be returned as a result of the UPDATE DATAGRP command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 327. Return and reason codes for the UPDATE DATAGRP command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE DATAGRP command completed successfully.
X'00000008'	X'00002004'	An invalid keyword or more than one keyword is specified on the UPDATE DATAGRP command.
X'00000008'	X'00002014'	The UPDATE DATAGRP command is not processed because an invalid character is found in the data group name.
X'00000008'	X'00002040'	More than one filter or keyword value is specified on the UPDATE DATAGRP command. Either more than one keyword or an invalid combination of filters was specified. For example, START(ACCESS) and STOP(ACCESS) was specified, or OPTION(DBALLOC,NODBALLOC) was specified. Check the input command and reenter the correct combinations.
X'00000008'	X'00005000'	The UPD DATAGRP command processing terminated because IMODULE GETSTOR storage could not be obtained.

Table 327. Return and reason codes for the UPDATE DATAGRP command (continued)

Return code	Reason code	Meaning
X'00000008'	X'0000501C'	The UPD DATAGRP command processing terminated because IMODULE GETMAIN storage could not be obtained.
X'0000000C'	X'00003000'	The UPDATE DATAGRP command is successful for at least one member in the data group. The UPDATE DATAGRP command is not successful for one or more members in the data group. The completion code indicates the reason for the error with the data group member name. The completion codes that can be returned by the UPDATE DATAGRP command are listed in the UPDATE DATAGRP completion code table.
X'0000000C'	X'00003004'	The UPDATE DATAGRP command is not successful for all the members in the data group. The completion code indicates the reason for the error with the data group member name. The completion codes that can be returned by the UPDATE DATAGRP command are listed in the UPDATE DATAGRP completion code table.
X'00000010'	X'0000400C'	The UPDATE DATAGRP command is not processed because it is not valid on the XRF alternate.
X'00000010'	X'00004014'	The UPDATE DATAGRP command is not processed because it is not valid on the RSR tracker.
X'00000010'	X'00004200'	Commands are not processed because IMS shutdown is in progress.
X'00000010'	X'00004400'	MINVERS in the RECON data sets is not 11.1.
X'00000014'	X'00005004'	The UPDATE DATAGRP command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	The UPDATE DATAGRP command processing terminated because DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	The UPDATE DATAGRP command processing terminated because AWE could not be obtained.
X'00000014'	X'00005014'	The UPDATE DATAGRP command processing terminated because BCB storage could not be obtained.
X'00000014'	X'00005FFF'	The UPDATE DATAGRP command processing terminated because of an internal error.
X'02000008'	X'0000203C'	An invalid TIMEOUT value is specified. The value must be a numeric value between 1 and 999.

The following table includes an explanation of the completion codes. Errors unique to the processing of UPDATE DATAGRP command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 328. Completion codes for the UPDATE DATAGRP command

Completion code	Completion code text	Meaning
0		The command completed successfully for the resource.
8	COMMAND COMPLETE FOR SOME	Some. The command completed with error for some of the AREAS of the DEDB. Response lines for the area names in error are returned.
C	COMMAND COMPLETE FOR NONE	None. The command completed with error for all the AREAs of the DEDB. Response lines for the area names in error are returned.
10	NO RESOURCES FOUND	No resource found. Database name is invalid, or the wildcard parameter specified does not match any database names.
11	DUPLICATE RESOURCE NAME	Duplicate resource name. The resource name is specified multiple times on the command and is ignored.
17	ANOTHER CMD IN PROGRESS	This error is returned when the quiesce could not be started because another database command was in progress.
23	DB STOP ACCESS IN PROGRESS	A /DBRECOVERY, or UPDATE DB STOP(ACCESS) command to stop database access is in progress. This takes the database offline.
25	DB STOP UPDATES IN PROGRESS	A /DBDUMP or UPDATE DB STOP(UPDATES) command to stop database updates is in progress.
26	DEDB STOP IN PROGRESS	/DBRECOVERY, /STOP, or UPDATE DB STOP(SCHD) command to stop database scheduling is in progress for a DEDB.
31	NOT ALLOWED FOR A DEDB	Database is a DEDB. The command entered is not valid for the DEDB in the IMS environment.
32	NOT ALLOWED FOR AN MSDB	Database is a MSDB. The command entered is not valid for the MSDB in the IMS environment.
33	NOT ALLOWED FOR A HALDB MASTER	Command invalid HALDB master. The command OPTION is invalid for the HALDB master but partition structure rebuild will be done if structure rebuild is needed and if only one HALDB master was specified in the command. No rebuild will be attempted if there is more than one database name listed in the command. If there are multiple database names listed in the command and all are invalid except the HALDB master, then rebuild will be attempted if needed.
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for an IMS descriptor or resource. DFSDSDB1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).

Table 328. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
53	NO RM ADDRESS SPACE	This error is returned when the quiesce could not be started because the RM address space is not present.
55	NO FAST PATH INSTALLED	The command failed because Fast Path is not installed.
56	FF DB + LSO=Y + TRK = ERROR	Command is invalid on the RSR tracker because of the LSO=Y option.
65	DMB POOL STORAGE ERROR	DMB pool storage error. The command failed because of DMB pool storage request failure.
66	DMB POOL FULL	DMB pool full. The command failed because the DMB pool was full.
6C	NOT ALLOWED FOR A HALDB PARTITION	An UPDATE command specified a change to the residency option for a HALDB partition. The residency option is valid only for the master and not the partitions.
6F	REFERENCED BY PROGRAM	An UPDATE DB command is issued to change the resident option. There is a currently scheduled program that is referencing that database. The UPD command fails.
76	RECOVER CMD ACTIVE	/RECOVER START command is in progress to recover one or more databases with the database recovery services.
81	DBRC ERROR	DBRC error.
90	INTERNAL ERROR	The command entered is not processed because of an internal error.
91	TIMEOUT ERROR	This error is returned when the quiesce could not be completed within the timeout period.
92	COMMAND PROCESSING ERROR	Command processing error. The command entered is not processed because of a command error. A unique completion code could not be generated to explain the error. The message number and the return code that could not be converted to a completion code are listed in the error text.
A0	DYNAMIC ALLOCATION FAILED	Dynamic allocation failed. The command entered is not processed because the dynamic allocation failed for the DB.
A1	DB IS AUTHORIZED BY BATCH	Database is authorized by Batch. The command entered is not processed because the database is authorized by batch.
A2	DB IS AUTHORIZED BY ANOTHER IMS	Database is authorized by another IMS. The command entered is not processed because the database is authorized by another active or abnormally terminated IMS and its authorization state is incompatible with the current authorization request.
A3	AUTHORIZATION CHANGE FAILED	Authorization change failed. The DBRC CHNGAUTH request resulted in an error.

Table 328. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
A4	DATABASE NOT REGISTERED TO DBRC	Database not registered to DBRC. The command processing failed as the database is not registered to DBRC.
A5	PREVENT FURTHER AUTH ON	Prevent Further Auth ON. The command entered is not processed because the database or area is defined to DBRC as prevent further Auth.
A6	INVALID DATABASE RECORD IN RECON	Invalid database record in RECON. The command entered is not processed because an invalid parameter was found during the evaluation process of the database usage compatibility. The database record might be invalid in the RECON data set.
A7	DBRC UNAUTH FAILED FOR CHNGAUTH	DBRC unauth failed during change authorization. The command is not processed because of an error during UNAUTH processing during change authorization request.
A8	INVALID DB RECORD IN RECON	Invalid database record in RECON. An UPDATE DB SET(ACCTYPE) command is entered to change the database authorization level. An encoded state of zero is returned by DBRC during the change authorization processing.
A9	DB OR AREA AUTHORIZATION ERROR	Database or area authorization error. For a database, the command entered is not processed because of a database authorization error. For an area, area authorization to DBRC failed.
AA	DB IN USE-BMP	The UPDATE DB command is rejected because the database is in use by a BMP.
AB	DB IN USE-DBCTL LONG THREAD	The UPDATE DB command is rejected because the database is in use by a long-running DBCTL thread.
AC	FP AREA HELD-LONG BUSY WAIT	The UPDATE DB command is rejected because the area of the DEDB is in long-busy wait.
AD	DYNAMIC UNALLOCATION FAILED	The UPDATE DB command is not successful because of a dynamic unallocation error.
AE	DYNAMIC ALLOCATION ERROR	The UPDATE DB command is not successful because of a dynamic allocation error. No SVC99 is issued.
C1	OLR DDIR MISSING OR DFSPNT ZERO	Unknown DMB referenced for database. The command cannot be processed because an unknown data management block is referenced for the database. Refer to the DFS564I message put out to the system console to identify the DMB name that cannot be referenced.
CC	OLR IS ACTIVE FOR DATABASE	OLR is active for database. The command failed as OLR is active for the database.
D0	DATABASE CLOSE ERROR	Database close error. The command processing failed because of a database close error.

Table 328. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
D1	DATABASE WRITE ERROR	Database write error. The command processing failed because of a database write error.
D2	DATABASE NEEDS BACKOUT	Database needs backout. The command processing failed as the database needs backout.
D3	DATABASE OR AREA NEEDS RECOVERY	Database or AREA needs recovery. The command processing failed as the database or area needs recovery.
D4	DATABASE NEEDS IMAGE COPY	Database needs image copy. The command processing failed as the database needs image copy.
D5	DATABASE HAS NO BACKOUTS	Database has no backouts. The command processing failed as there are no backouts for the database.
D6	DATABASE IN USE	Database in use. A SET(ACCTYPE) is specified for the DEDB and the authorization level cannot be changed as the DEDB is in use in a region.
D7	DB I/O PREVENTION NOT COMPLETE	Database I/O prevention not complete. The database cannot be started as it is extended because of a XRF takeover and the I/O prevention is not complete.
D8	DATABASE BACKOUTS PENDING	Database backouts pending. The access type specified for the database cannot be changed as restartable backouts are pending for the database.
D9	DATABASE/AREA OPEN FAILED	Database or area open failed. The command failed because an error occurred while opening the database or area.
DA	DATABASE BEING RECALLED BY HSM	Database being recalled from HSM. The command processing failed because the database is being recalled from HSM.
DB	PARTITION OPEN FAILED	Partition open failed. The partition open failed because the master is offline. This can also occur if the partition has been deleted and partition structure rebuild has occurred. Partition structure rebuild can be accomplished by issuing an UPD DB NAME(haldbmst) START(ACCESS) OPTION(OPEN) command, where haldbmst is the partition's master, or by issuing a qualified GU call for a key in the key range of the partition. List.recon can be used to determine if the partition exists or has been deleted.
DC	HALDB PARTITION BUILD FAILURE	Database partition build failure. The database partition build for the DDIR or DMB failed. Refer to the DFS0415I message sent to the system console to determine the reason of the failure.
DD	HALDB PARTITION INIT FAILURE	Database partition initialization failed. The database partition initialization for the DDIR or DMB failed. Refer to the DFS0415 message sent to the system console for the details.

Table 328. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
DE	ACBLIB READ FAILURE	ACBLIB read failure. The command is not processed because there was an error reading the ACBLIB.
DF	DB DIRECTORY INIT FAILURE	Database directory initialization failed. The command is not processed because of a database directory initialization failure.
E0	DATABASE OR AREA IN RECOVERY	Database or area in recovery. The command is not processed because the database or area is in recovery.
E1	DATABASE HAS NOT BEEN DBR'ED	An UPDATE command changing the residency option of a database was not issued for a database that has not been DBR'ed.
E2	PARALLEL DB OPEN NOT COMPLETE	Restart parallel DB open not complete. The command is not processed because the restart parallel DB open is not complete for the database.
E5	PARTICIPANT UNABLE TO QUIESCE	This error is returned when the quiesce could not be completed successfully across the IMSplex. The IMS with this completion code was the quiesce participant which was not able to be quiesced.
E6	QUIESCE COMMUNICATION FAILURE	This error is returned when the quiesce could not be completed because of a failure to communicate across the IMSplex. There could be a problem with RM, OM, or SCI that has caused the communication failure to occur.
E7	CMD NOT ALLOWED	This error is returned when the database command could not be processed because a quiesce command was in progress.
E8	DATABASE HAS INTENT TO REORGANIZE	This error is returned when the quiesce could not be started because the RECON data sets indicates that there is intent to reorganize the database.
E9	DB IN WRONG STATE TO BE QUIESCED	This error is returned when the named resource is in the wrong state for quiesce processing to proceed.
EE	DATABASE BACKOUT ERROR	Database backout error. The command processing failed because of a database backout error.
EF	DATABASE IS IN ERROR	Database is in error. The command entered is not processed because the database is in error.
F0	NO AREA LOCK	No AREA lock. The command processing failed because the area lock could not be obtained.
F1	AREA NOT STOPPED	Area not stopped. The command entered is not processed because the AREA is not stopped.
F2	PRELOAD IS ACTIVE FOR AREA	Preload is active for AREA. The command is not processed because preload is active for the AREA.

Table 328. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
F3	UNRESOLVED INDOUBTS FOR AREA	Unresolved indoubts for AREA. The command entered is not processed because unresolved indoubts exist for the AREA.
F4	ALLOCATION FAILED	Allocation failed. Allocation failed for the AREA name.
F5	AREA NEEDS RECOVERY	AREA needs recovery. The command processing failed because the area needs recovery.
F6	ADS NUMBER DISCREPANCY	ADS number discrepancy. The command failed for the AREA because there is a discrepancy between the number of ADS allocated by IMS and the number of ADS known to DBRC. Correct the discrepancy and reissue the command to deallocate the data sets.
F7	AREA IS NOT LOADED INTO CF	Area is not loaded into CF. The AREA OPEN failed and is not loaded into the Coupling Facility.
F8	AREA HAS I/O TOLERATED CI	Area has I/O tolerated CI. The command is not processed because the AREA has an I/O tolerated CI.
F9	AREA HAS 2ND CI EEQE	Area has second CI EEQE. The command is not processed because the AREA has a second CI EEQE.
FC	UTILITY ACTIVE ON AREA	This error is returned when the area is in use by a utility.
FD	AREA HAS EEQE	This error is returned when the area has an extended error queue element (EEQE).
FE	AREA HAS EQE	This error is returned when the area has an error queue element (EQE).

Examples

The following are examples of the UPDATE DATAGRP command:

Example 1 for UPDATE DATAGRP command

TSO SPOC input:

```
UPD DATAGRP NAME(GROUP1) STOP(ACCESS)
```

TSO SPOC output:

```
DataGroup DBName   AreaName MbrName    CC
GROUP1                      IMSA         0
GROUP1                      IMS1         8
GROUP1    DEDBJN03          IMS1         C
GROUP1    DEDBJN03 DB3AREA0 IMS1        F0
```

OM API input:

```
CMD(UPD DATAGRP NAME(GROUP1) STOP(ACCESS))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.129 22:45:55.898843</statime>
<stotime>2003.129 22:45:55.990001</stotime>
<staseq>B965346AB45DB229</staseq>
<stoseq>B965346ACA9F1263</stoseq>
<rqsttkn1>USRT005 10154555</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>

</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntext>At least one request successful</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMSA    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>DATAGRP         </kwd>
<input>UPD DATAGRP NAME(GROUP1) STOP(ACCESS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DG" llbl="DataGroup" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DG(GROUP1  ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>DB(DEDBJN03) AREA(DB3AREA0) DG(GROUP1  ) MBR(IMS1    ) CC(  F0)
</rsp>
<rsp>DB(DEDBJN03) DG(GROUP1  ) MBR(IMS1    ) CC(   C) </rsp>
<rsp>DG(GROUP1  ) MBR(IMS1    ) CC(   8) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The command stops access to all the members of data group GROUP1 and takes them offline. The command response lines are returned for all the databases or areas that resulted in an error. No response lines are returned for members for which the command was successful.

Example 2 for UPDATE DATAGRP command

TSO SPOC input:

```
UPD DATAGRP NAME(GROUP1) START(ACCESS) OPTION(ALLRSP)
```

TSO SPOC output:

```
DataGroup DBName   AreaName MbrName    CC
GROUP1                      IMSA        0
GROUP1            DB21AR1   IMSA        0
GROUP1    BE3PARTS          IMSA        0
GROUP1    DEDBJN03          IMSA        0
GROUP1                      IMS1        0
GROUP1            DB21AR1   IMS1        0
GROUP1    BE3PARTS          IMS1        0
GROUP1    DEDBJN03          IMS1        0
```

OM API input:

```
CMD(UPD DATAGRP NAME(GROUP1) START(ACCESS) OPTION(ALLRSP))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.129 22:46:51.490484</statime>
<stotime>2003.129 22:46:51.672332</stotime>
<staseq>B965349FB88B4445</staseq>
<stoseq>B965349FE4F0C36A</stoseq>
<rqsttkn1>USRT005 10154651</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMSA    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>DATAGRP         </kwd>
<input>UPD DATAGRP NAME(GROUP1) START(ACCESS) OPTION(ALLRSP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DG" llbl="DataGroup" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(DEDBJN03) DG(GROUP1  ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>DB(BE3PARTS) DG(GROUP1  ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>AREA(DB21AR1 ) DG(GROUP1  ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>DG(GROUP1  ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>DB(DEDBJN03) DG(GROUP1  ) MBR(IMS1    ) CC(   0) </rsp>
<rsp>DB(BE3PARTS) DG(GROUP1  ) MBR(IMS1    ) CC(   0) </rsp>
<rsp>AREA(DB21AR1 ) DG(GROUP1  ) MBR(IMS1    ) CC(   0) </rsp>
<rsp>DG(GROUP1  ) MBR(IMS1    ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The command starts access of all the members of data group, GROUP1, and makes them available. The command response lines are returned for all the databases or areas that are processed with OPTION(ALLRSP) specified.

Example 3 for UPDATE DATAGRP command

TSO SPOC input:

```
UPDATE DATAGRP NAME(DBGXYZ) START(QUIESCE) OPTION(HOLD)
```

TSO SPOC output:

```
DataGroup  DBName   MbrName    CC
DBGXYZ     DB1XYZ   IM02        0
DBGXYZ     DB2XYZ   IM02        0
DBGXYZ              IM02        0
DBGXYZ     DB1XYZ   IM01        0
DBGXYZ     DB2XYZ   IM01        0
DBGXYZ              IM01        0
DBGXYZ     DB1XYZ   IM03        0
DBGXYZ     DB2XYZ   IM03        0
DBGXYZ              IM03        0
```

Explanation: This example is of a successful quiesce-and-hold for a data group.

Example 4 for UPDATE DATAGRP command

TSO SPOC input:

```
UPD DATAGRP NAME(DBGXYZ) START(QUIESCE)
```

TSO SPOC output:

```
DataGroup  DBName   MbrName    CC
DBGXYZ     DB1XYZ   IM02        0
DBGXYZ     DB2XYZ   IM02        0
DBGXYZ              IM02        0
DBGXYZ     DB1XYZ   IM01        0
DBGXYZ     DB2XYZ   IM01        0
DBGXYZ              IM01        0
DBGXYZ     DB1XYZ   IM03        0
DBGXYZ     DB2XYZ   IM03        0
DBGXYZ              IM03        0
```

Explanation: This example is of a successful quiesce-and-go for a data group.

Example 5 for UPDATE DATAGRP command

TSO SPOC input:

```
UPD DATAGRP NAME(DBGXYZ)  STOP(QUIESCE)
```

TSO SPOC output:

```
DataGroup  DBName   MbrName    CC
DBGXYZ     DB1XYZ   IM02        0
DBGXYZ     DB2XYZ   IM02        0
DBGXYZ              IM02        0
DBGXYZ     DB1XYZ   IM01        0
DBGXYZ     DB2XYZ   IM01        0
DBGXYZ              IM01        0
DBGXYZ     DB1XYZ   IM03        0
DBGXYZ     DB2XYZ   IM03        0
DBGXYZ              IM03        0
```

Explanation: This example is of releasing the quiesce for a data group.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

"/START DATAGRP command" on page 461

"/STOP DATAGRP command" on page 515

UPDATE DB command

Use the UPDATE DB command to update status or definition information about databases. For example, UPDATE DB can make a database available, take the database offline, stop scheduling, stop updates, lock, and unlock a database.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 661
- "Usage notes" on page 670
- "Equivalent IMS type-1 commands" on page 672
- "Output fields" on page 672
- "Return, reason, and completion codes" on page 673
- "Examples" on page 683

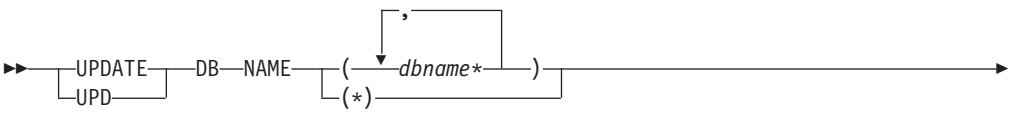
Environment

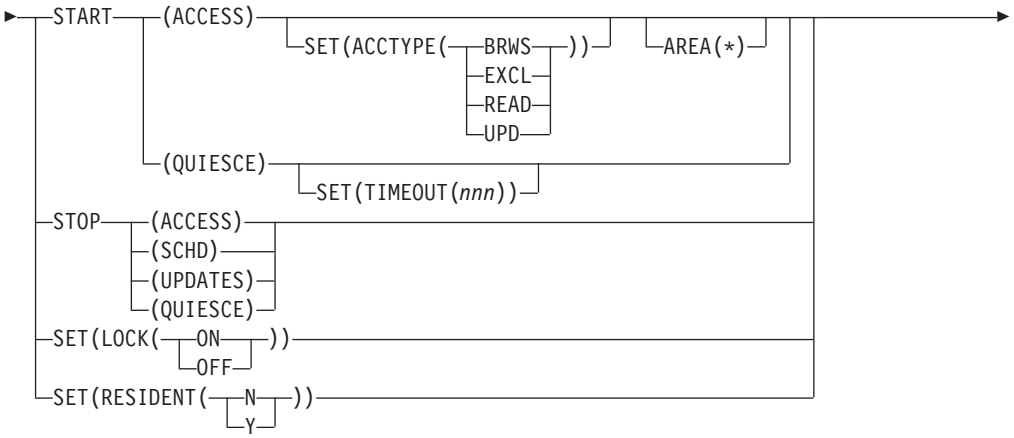
The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE DB command and keywords can be issued.

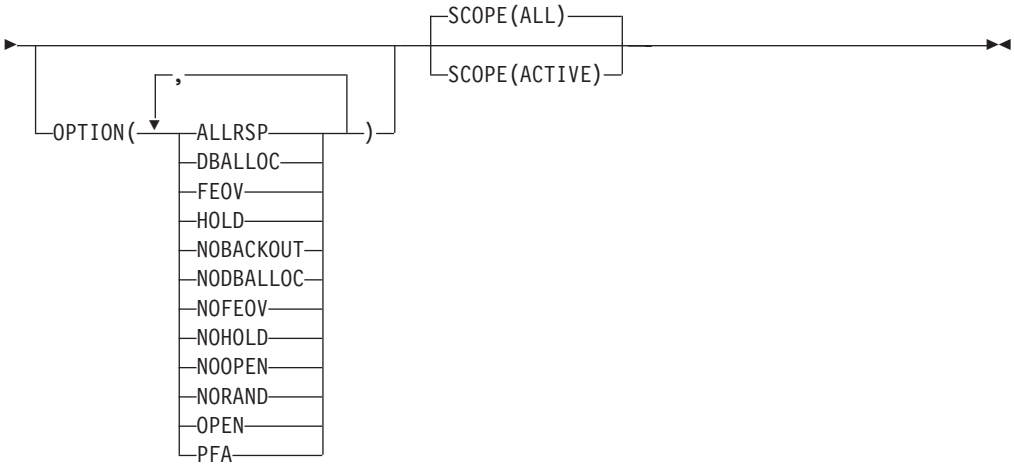
Table 329. Valid environments for the UPDATE DB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE DB	X	X	
NAME	X	X	
OPTION	X	X	
SCOPE	X	X	
SET	X	X	
START	X	X	
STOP	X	X	

Syntax







Keywords

The following keywords are valid for the UPDATE DB command:

AREA(*)

Specifies the areas of the databases that are to be processed or, if you specify AREA(*), indicates that the command applies to all the areas in the database.

The UPDATE DB START(ACCESS) AREA(*) command starts the DEDB and all of its areas and allows scheduling of new applications against the DEDB.

The AREA(*) keyword is ignored for non-DEDB databases. You cannot specify both the AREA(*) and the NAME(*) keywords. If you specify the AREA(*) keyword with the START(ACCESS) SET(ACCTYPE) keyword, all the DEDB areas are restarted after they are stopped by the START(ACCESS) SET(ACCTYPE) keyword.

NAME()

Specifies the 1-8 character name of the database (DBD name). Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command

responses are returned for all the resource names that are processed. For NAME(*) command responses are returned only for the resource names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the resource names that are processed.

NAME(*) is the designed method on IMSplex commands to enable the user to specify that the command applies to all the database resources.

The database names that match the generic or wildcard parameter are processed. For specific or wildcard names, response lines are returned for all the database names that are processed.

When the database specified is a DEDB, a response line is returned for the DEDB name. A response line is also returned for all the areas of a DEDB that resulted in errors. A response line is not returned for the areas of a DEDB for which the command action is successful.

OPTION()
Specifies additional functions to be performed along with the UPDATE DB command.

ALLRSP
Indicates that the response lines are to be returned for all resources that are processed on the command. The default action is to return response lines only for the resources that resulted in an error. It is only valid with NAME(*) or AREA(*). ALLRSP is ignored for other NAME values.

DBALLOC
Indicates that the database is to be allocated when it is started.

DBALLOC can be specified with START(ACCESS) only. DBALLOC is the default action when specific database names are specified on the command. Only one of DBALLOC or NODBALLOC can be specified.

OPTION(DBALLOC) is not valid for a HALDB master.

FEOV Indicates to force end of volume after the command processing is complete.

The IMS log switches to the next OLDS, and a simple checkpoint is taken unless specified with STOP(QUIESCE). This switch is marked as a recovery point for log archiving purposes. A simple checkpoint is not taken when specified with STOP(QUIESCE).

FEOV is valid only with START(QUIESCE) OPTION(NOHOLD), STOP(QUIESCE), STOP(UPDATES), or STOP(ACCESS), and does not apply to an RSR tracker. OPTION(FEOV) is the default when specified with START(QUIESCE) OPTION(NOHOLD) or STOP(QUIESCE).

When START(QUIESCE) or STOP(QUIESCE) is specified, OPTION(FEOV) is the default; that is, the end of volume will be forced after the quiesce point has been reached. The logs are switched before the quiesce is released so that any new updates occur on the new IMS log. Switching of the logs occurs when STOP(QUIESCE) is issued to release a quiesce from a previous START(QUIESCE) OPTION(HOLD).

HOLD
Specifies that after the quiesce has been achieved successfully, the database should remain quiesced. A subsequent STOP(QUIESCE) would be required to release the quiesce on the database.

OPTION(HOLD) is valid only with the START(QUIESCE) keyword. OPTION(HOLD) is mutually exclusive with OPTION(NOHOLD).

NOBACKOUT
Indicates to suppress backout restart for a database not registered in DBRC.

NOBACKOUT can be specified with START(ACCESS) only and is not valid with NAME(*). OPTION(NOBACKOUT) is not valid for a HALDB master.

NODBALLOC
Indicates that the database is not to be allocated when it is started.

The database is allocated when it is scheduled. NODBALLOC can be specified with START(ACCESS) only. NODBALLOC is the default action when NAME(*) is specified. Only one of DBALLOC or NODBALLOC can be specified. OPTION(NODBALLOC) is ignored for HALDB masters.

NOFEOV
Indicates to not force end of volume after the command processing is complete. The IMS log does not switch to the next OLDS and a simple checkpoint is not taken.

OPTION(NOFEOV) can be specified only with START(QUIESCE) OPTION(NOHOLD), STOP(ACCESS), STOP(QUIESCE), or STOP(UPDATES). OPTION(NOFEOV) is the default except when START(QUIESCE) or STOP(QUIESCE) is specified.

NOHOLD
Specifies that after the quiesce has been achieved successfully the database should automatically release the quiesce. If either HOLD or NOHOLD is not specified, NOHOLD is assumed.

OPTION(NOHOLD) is valid only with the START(QUIESCE) keyword. OPTION(NOHOLD) is mutually exclusive with OPTION(HOLD).

NOOPEN
Indicates that the database is not to be opened when it is started.

NOOPEN can only be specified with START(ACCESS). NOOPEN is the default action unless the database has EEQE's.

Only one of OPEN or NOOPEN can be specified. OPTION(NOOPEN) is not valid for a HALDB master.

NORAND
Indicates that the randomizer stays loaded while the UPDATE DB STOP(ACCESS) command is being processed.

This option solves the problem of the Extended Common Service Area (ECSA) becoming fragmented by continuous unloading and reloading of randomizers.

Restriction: OPTION(NORAND) only works for Fast Path DEDBs. It is ignored for full-function databases.

OPTION(NORAND) status is not maintained in the RM resource structure if global database status is maintained. If an UPD DB STOP(ACCESS) OPTION(NORAND) command is issued for a Fast Path DEDB, the STOACC status is saved in the RM resource structure. The randomizer is unloaded when the global STOACC status is applied.

OPEN Indicates that the database is to be opened when it is started. OPEN can be specified only with START(ACCESS).

OPEN cannot be specified with NAME(*) or if NODBALLOC is specified. Either OPEN or NOOPEN can be specified. OPTION(OPEN) is not valid for a HALDB master, but the partition structure rebuild is done if structure rebuild is needed and if only one HALDB master was specified in the command. No rebuild is attempted if there is more than one database name listed in the command.

The UPDATE DB AREA(*) START(ACCESS) OPTION(OPEN) command starts and opens all areas under one or more DEDBs specified in the NAME() keyword even if the areas are not registered on DBRC as PREOPEN. The OPTION(OPEN) is processed locally by all IMS systems that receive the command and is not maintained as a global status in RM.

PFA Sets or resets the "prevent further authorization" (PFA) or the "read-only" status in the RECON for the database. Use the PFA option with UPDATE DB START(ACCESS) to enable access to a database. Use the PFA option with the UPDATE DB STOP (ACCESS|SCHD|UPDATES) command to prevent access to a database. PFA is optional. If you specify OPTION(PFA), the command master updates the RECON data set only if the command is successful at the master IMS. All other IMS systems that receive the command process is locally. OPTION(PFA) is not valid with SET(LOCK(ON|OFF)).

SCOPE()

Specifies where IMS should apply the change.

ALL

Changes are applied to the IMS systems that are currently active and to which the command is routed. Changes are also applied globally by updating the value maintained by RM. The global status that RM saves can be propagated across some IMS restarts. When an IMS system starts, it obtains global status information from RM. Maintaining global status in RM allows IMS systems to start without a specific setting or status for an area.

If the status or attribute that this command is updating is not maintained globally by RM, then the command is processed as if SCOPE(ACTIVE) is specified. If this command updates status or attributes that are both global and local, RM only updates the global status or attributes.

You must specify if the global area status must be maintained in RM. You can specify this during IMS initialization in either the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSDB(Y)). You

can also change it dynamically using the UPD IMS SET(PLEXPARM(GSTSDB(Y)) command. If you do not specify that global database status is to be maintained, the GLOBAL keyword is processed as in prior releases, and the global status is not updated.

Restriction: SCOPE(ALL) does not apply to the quiesce function. The scope of a quiesce is always all instances of the database usage across the IMSplex. There is not a global status of QUIESCE for a database. For START(QUIESCE) and STOP(QUIESCE), SCOPE(ALL) is the same as SCOPE(ACTIVE).

ACTIVE
Changes are applied to the active IMS systems to which the command is routed to. Any global status information maintained in the RM resource structure is not changed by the SCOPE(ACTIVE) command.

Restriction: SCOPE(ACTIVE) does not apply to the quiesce function. The scope of a quiesce is always all instances of the database usage across the IMSplex.

RM maintains global status information for some database attributes and values. IMS updates those values in RM based on commands or other internal events. When SCOPE(ALL) is specified, every IMS system that processes the command updates information in its local control blocks. Only the IMS command master calls RM to update the information globally.

The UPDATE DB command keywords that update information locally in the active IMS system and globally in RM when SCOPE(ALL) is specified include:

- START(ACCESS)
- START(ACCESS) SET(ACCTYPE())
- STOP(ACCESS|SCHD|UPDATES)
- SET(LOCK(ON|OFF))

If global status is maintained, UPD DB START(ACCESS) STOP(ACCESS,SCHD,UPDATES) SCOPE(ALL) writes an X'4C' log record to include global status and global time stamp.

SET() Specifies the attribute values to be changed or sets the database state.

ACCTYPE
Specifies the access intent for the named database. ACCTYPE() can only be specified if START(ACCESS) is also specified. This keyword overrides the database access intent specified at system definition.

BRWS The database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those databases that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data since a sharing IMS subsystem could be updating the database. The database is opened for read-only processing.

EXCL The database is to be used exclusively by this IMS

subsystem. This exclusive access is guaranteed only when the database is registered to DBRC.

READ The database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. Access type READ differs from access type BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UPD The database is for update as well as read processing in the IMS subsystem.

RESIDENT

Specifies the resident option. The RESIDENT(N) option takes effect right away. The RESIDENT(Y) option takes effect at the next restart, unless the database was updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart. A database defined as a DEDB in ACBLIB always sets the RESIDENT(Y) attribute when the DEDB is loaded, regardless of the RESIDENT value specified. RESIDENT(N) is rejected for a DEDB.

N The DMB associated with the named database resource is not made resident in storage. The DMB is loaded at scheduling time.

Y The DMB associated with the named database resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the DMB and initializes it. A resident database is accessed in local storage, which eliminates I/O to the ACBLIB. In an online environment, the DMB control blocks are stored in the ACBLIB. If the DLI/SAS address space exists, DLI/SAS loads the DMB, otherwise, it is the IMS control region that loads it. This makes the DMB dependent on the existence of the corresponding database resource.

The database must be stopped in order for this attribute to be changed. You may need to issue a /DBR DB command or an UPDATE DB STOP(ACCESS) command to stop the database before issuing an UPDATE DB SET(RESIDENT(Y|N)) command.

An UPDATE command specified with SET(RESIDENT) is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined). The UPDATE DB command changes a MODBLKS database to dynamic, if the RESIDENT attribute is changed. These commands are recoverable.

LOCK Locks and unlocks the specified database. NAME(*) cannot be specified with SET(LOCK(ON|OFF)).

ON Locks and prevents subsequently scheduled programs from accessing the database, without affecting currently scheduled programs. The database is not closed.

For a shared secondary index database, an UPDATE DB SET(LOCK(ON)) on the first secondary index or subsequent secondary indexes affects only the named database.

OFF Unlocks the specified databases and resets the effect of an UPDATE DB SET(LOCK(ON)) command. An UPDATE DB SET(LOCK(OFF)) on the first secondary index or subsequent secondary indexes affects only the named database.

TIMEOUT(*nnn*)
Specifies the number of seconds to wait before a timeout occurs in a database quiesce. The timeout value can be 1 - 999 seconds. The TIMEOUT parameter value can override the DBQUIESCETO parameter in the DFSCGxxx member of the IMS PROCLIB data set. If the TIMEOUT parameter is omitted and the DBQUIESCETO parameter is not specified, the default timeout value is 30 seconds. The TIMEOUT keyword is valid only with the START(QUIESCE) keyword.

START()
Specifies the attributes that are to be started.

ACCESS
The UPDATE DB START(ACCESS) command starts the database and permits access from transactions or programs. The UPDATE DB START(ACCESS) command resets the actions done by a prior UPDATE DB STOP(SCHD), or UPDATE DB STOP(UPDATE). The access intent is set to the database access intent specified at system definition unless SET(ACCTYPE) is specified.

The UPDATE DB START(ACCESS) command can be used to allocate or reallocate all databases other than DEDBs. For a DEDB, an UPDATE AREA command can be used to allocate or reallocate the DEDB areas.

For a DEDB, the UPDATE DB START(ACCESS) command also causes any unloaded randomizer that was specified in the DBD source to be reloaded.

When the UPDATE DB START(ACCESS) command is specified for a HALDB partition, the partition is not allocated unless it has EEQEs, the OPEN keyword is specified, or it was previously authorized but not allocated. The action taken to allocate the data sets is dependent on the status of the master database and its availability.

The UPDATE DB START(ACCESS) command may reset the USTOPPED status for transactions that are suspended. If the transaction is suspended and its processing program has access to the started database, the UPDATE DB START(ACCESS) command results in the USTOPPED attribute being reset. Any messages on the suspend queue for that transaction is transferred to the normal processing queue.

If there was a prior dynamic backout or emergency restart backout failure, then the UPDATE DB START(ACCESS) command attempts to perform the backout again.

If the database is registered to DBRC, then DBRC is informed when batch backout is successfully executed, and the failing backout is not attempted again when an UPDATE DB START(ACCESS) command is issued.

The UPDATE DB START(ACCESS) command is not processed for the databases being accessed by batch programs.

For a HIDAM database, the UPDATE DB START(ACCESS) command must be issued for both the index and the data area DBD. If a backout failure occurs for this database, the command causes the backout to be attempted again.

On an RSR tracker, the UPDATE DB START(ACCESS) command can be used to resume tracking for those databases that were stopped by a tracking subsystem processing. The command can also be used to start online forward recovery (OFR) for those databases that are not current with mainline tracking.

For shared secondary index databases, the UPDATE DB START(ACCESS) can be issued on the first secondary index or subsequent secondary indexes to undo the actions of the prior UPDATE DB STOP(SCHD) and UPDATE DB STOP(UPDATES) commands.

Additional functions can be performed with START(ACCESS) by specifying the OPTION keyword. OPTION(DBALLOC | NODBALLOC), OPTION(NOBACKOUT), OPTION(NOOPEN | OPEN) can be specified along with START(ACCESS). The OPTION keyword is not valid for a HALDB master.

QUIESCE

Specifies that the databases named on the command are to be quiesced to establish a new recovery point. The scope of a quiesce is always all instances of the database usage across the IMSplex. There is no quiesce that would apply only to a subset of the IMSplex.

STOP()

Specifies the attributes that are to be stopped.

ACCESS

Starts offline processing of the database. This processing closes and deallocates the database and deauthorizes the database to DBRC. An UPDATE DB START(ACCESS) command is required to reset the effect of an UPDATE DB STOP(ACCESS).

OPTION(NOFEOV) is the default action for STOP(ACCESS). The IMS log is not switched to the next OLDS and a simple checkpoint is not taken. OPTION(FEOV) can be specified on the command to switch to the next OLDS and take an IMS simple checkpoint.

The UPDATE DB STOP(ACCESS) command can be used on a database readiness level (DLT) tracker to take shadow areas and databases offline for image copy and recovery. The command can also be used to stop online forward recovery (OFR) in progress for the specified database.

The UPDATE DB STOP(ACCESS) command does not deallocate a data set if a VSAM data set hardware error occurred. For shared secondary index databases, an UPDATE DB STOP(ACCESS) on the first secondary index affects all databases sharing the secondary index data set. An UPDATE DB STOP(ACCESS) command on the subsequent secondary indexes affects only the named database.

QUIESCE
Specifies that the databases named on the command should be made available again by releasing the quiesce on the databases.

Unlike START(QUIESCE), where each database resource listed must be quiesced in order for the command to complete successfully, STOP(QUIESCE) continues to process each listed resource even if some resources are not in quiesced state or cannot be released from quiesced state. For those resources that are not in quiesced state or cannot be released from quiesced state, the command returns a response line for each of those resources.

SCHD Stops or prevents subsequently scheduled programs from accessing the database, without affecting currently scheduled programs. The database is not closed.

An UPDATE DB START(ACCESS) command can be used to reset the effect of an UPDATE DB STOP(SCHD) command.

If the command is issued for a DEDB or MSDB, programs using the database will not be scheduled. For other databases, the programs will still be scheduled. If the INIT call was issued, however, a call against the database will result in either a 3303 pseudoabend or a BA status code.

When the UPDATE DB STOP(SCHD) command is issued for a database that is in use by an MPP region, the command is processed after the region completes processing the current message. After the current message processing is complete, the application program receives a QC status indicating no more messages even if there are messages to be processed.

When the UPDATE DB STOP(SCHD) command is issued for a database that is in use by a BMP region, the command is rejected.

In a DBCTL system, CCTL can specify LONG or SHORT when it schedules a PSB. When the UPDATE DB STOP(SCHD) command is issued for a database that is in use by a LONG thread, the command is rejected. When the command is issued for a database that is in use by a SHORT thread, the thread completes before the command is processed.

For a shared secondary index database, an UPDATE DB STOP(SCHD) on the first secondary index or subsequent secondary indexes affects only the named databases.

UPDATES
Stops or prevents transactions or programs from updating the specified DL/I database.

STOP(UPDATES) is not valid for DEDBs or MSDBs.

An UPDATE DB START(ACCESS) command is required to reset the effect of an UPDATE DB STOP(UPDATES) command.

When the UPDATE DB STOP(UPDATES) command is processed, the message processing regions using the specified database are terminated at the conclusion of processing their transactions in preparation to close the database and enable the databases to be opened input only. As the message processing regions terminate programs, the data sets of the database are closed.

OPTION(FEOV) forces the IMS log to switch to the next OLDS. This switch is marked as a recovery point for log archiving purposes. IMS also issues a simple checkpoint. OPTION(NOFEOV), which is the default, overrides this action.

After the command is processed, the scheduling of transactions is resumed. No transactions will be allowed to update the specified databases. Programs with update intent will be scheduled, but update calls to DL/I databases will result in a 3303 pseudoabend, a BA, or BB status code. The pseudoabend or status codes appear only if the application program informed IMS through the `INIT STATUS GROUPA` or `GROUPB` call that it is prepared to accept status codes regarding data unavailability.

An UPDATE DB STOP(UPDATES) on the first secondary index affects all databases sharing the secondary index data set. An UPDATE DB STOP(UPDATES) on subsequent secondary indexes affects only the named database.

Usage notes

The UPDATE DB command can only be specified through the OM API and can only be processed by the DB/DC and DBCTL environments. When the UPDATE DB command is issued, it only applies to the IMS system to which it is routed. The command is not processed by other IMS systems in the IMSplex that share the database but do not receive the command. OM selects one IMS as the command master. Also, the command is not allowed on the XRF alternate system, RSR tracker, or FDBR region.

Resources exist for the life of IMS unless they are deleted using a DELETE command or online change for MODBLKS. Resource updates are recoverable across an IMS warm start or emergency restart. Updates to database runtime resource definition attributes such as RESIDENT are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

Each database is updated individually, unlike the online change process where either all databases are updated or no databases are updated. Some runtime resource definition values for a database can only be updated if the database is not in use. If the database is in use, the update fails. An exception to this rule is status. You can update the status of a database while it is in use. In a sysplex environment with multiple IMS systems, the update might succeed on some IMS systems and fail on others.

Runtime resource definition attributes include the following: RESIDENT.

An UPDATE DB command that changes runtime resource definition attributes such as the RESIDENT attribute is rejected for MSDBs and DEDBs.

The UPDATE DB command returns CCTXT with a nonzero completion code. The CCTXT can be up to 32 bytes, and it includes information about what the completion code means. The UPDATE DB SCOPE(ALL) command returns a response line with the completion code for the global status update. The CCTXT for that code is GBL CC.

If all the attributes specified by the UPDATE command are already defined for the resource, no update is actually made, no resources are quiesced, no log record is

created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB with the exception of the /START DATABASE and the UPDATE DB START(ACCESS) command. A HALDB partition is not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

The HALDB partition reflects conditions such as STOPPED (shown as STOACC or STOSCHD or STOUPDS on QUERY DB), LOCKED, or NOTOPEN. When a HALDB partition is stopped, it must be explicitly started again. The type-1 commands with the keyword ALL, type-2 commands with NAME(*), and commands against a HALDB master do not change the STOPPED (shown as STOACC, STOSCHD or STOUPDS on QUERY DB) and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is /DBR on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. On a /DISPLAY DB command, only the HALDB master displays STOPPED (each HALDB partition does not display STOPPED unless it was itself stopped). If a /DBR command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

Each partition inherits the access limitations of its HALDB master. If the /DBD command is issued against a HALDB master, all of its partitions close. A subsequent reference to any of the partitions results in the partition opening for input, although the partition's access might be UPDATE or EXCLUSIVE. The DBRC authorization state reflects the limited access.

If received during an UPD DB NAME(partname) START(ACCESS) command, where *partname* is a HALDB partition that was added and partition structure rebuild has not been done, then issue an UPD DB NAME(haldbmst) START(ACCESS) OPTION(OPEN), where *haldbmst* is the partition's master, followed by an UPD DB NAME(partname) START(ACCESS) command or issue a call for a key in the key range of the new or redefined partition. This will invoke partition structure rebuild and allow the partition to be used.

Restrictions:

- The UPDATE DB START(ACCESS | QUIESCE) and UPDATE DB STOP(ACCESS | QUIESCE | SCHD | UPDATES) commands cannot be processed against a HALDB partition on an IMS system while HALDB Online Reorganization (OLR) is running against that partition on the same IMS system.
- The UPDATE DB START(ACCESS) SET(ACCTYPE(UPD)) and UPDATE DB STOP(ACCESS | UPDATES) commands cannot be issued against a HALDB master while OLR is reorganizing any of its partitions.
- While the database is being quiesced, this command cannot be processed successfully.

When a nonzero return code is received for the UPDATE DB START(QUIESCE) OPTION(HOLD) command or the UPDATE DB STOP(QUIESCE) command, you

might need to correct the problem that is reported by the IMS reason code and completion code, and try the command again.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE DB command and the type-1 IMS commands that perform similar functions.

Table 330. Type-1 equivalents for the UPDATE DB command

UPDATE DB command	Similar IMS type-1 command
UPDATE DB NAME(*name*) STOP(UPDATES)	/DBD DB *dbname*
UPDATE DB NAME(*name*) STOP(ACCESS)	/DBR DB *dbname*
UPDATE DB NAME(*name*) START(ACCESS)	/START DB *dbname*
UPDATE DB NAME(*name*) STOP(SCHD)	/STOP DB *dbname*
UPDATE DB NAME(*name*) SET(LOCK(ON))	/LOCK DB *dbname*
UPDATE DB NAME(*name*) SET(LOCK(OFF))	/UNLOCK DB *dbname*

Output fields

The following table shows the output fields for the UPDATE DB. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 331. UPDATE DB output fields

Short label	Keyword	Meaning
AREA	*error*	Area name of the DEDB that resulted in an error during the processing of the command.
CC	N/A	Completion code.
CCTXT	*error*	The completion code text that briefly explains the meaning of the completion code.
DB	DB	Database name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
GBL	GBL	Indicates that the response line is for the global update.
MBR	N/A	IMSplex member that built output line. IMS identifier of the IMS for which the database information is displayed. IMS identifier is always returned.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the UPDATE DB command. The OM return and reason codes that may be returned as a result of the UPDATE DB command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 332. Return and reason codes for the UPDATE DB command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000008'	X'00002004'	An invalid keyword or filter, or more than one keyword is specified on the UPDATE DB command.
X'00000008'	X'00002008'	Invalid number of keywords. Either a SET, START, or STOP keyword is required.
X'00000008'	X'00002011'	UPDATE DB command processing terminated because you cannot specify more than one keyword with the NAME(*) keyword.
X'00000008'	X'00002014'	The UPDATE DB command is not processed because an invalid character is found in the database name parameter.
X'00000008'	X'00002040'	More than one filter or keyword value is specified on the UPDATE DB command. Either more than one keyword or an invalid combination of filters was specified. For example, UPDATE DB NAME(dbname) START(ACCESS) OPTION(NORAND) or UPDATE DB NAME(dbname) STOP(UPDATES) OPTION(NORAND)
X'00000008'	X'00002048'	Invalid SET attribute.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), output lines are only returned for resources with nonzero completion codes. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the UPDATE DB command are listed in the UPDATE DB completion code table.
X'00000010'	X'00004000'	The UPDATE DB command is not processed as restart is in progress.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.

Table 332. Return and reason codes for the UPDATE DB command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004024'	The UPDATE DB command is not processed because Fast Path is not installed.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004124'	An UPDATE DB command that specified either START(QUIESCE) or STOP(QUIESCE) was rejected, because another process of the same type, such as another command with QUIESCE specified, was already in progress. Wait until the other process has completed.
X'00000010'	X'000041F0'	The UPDATE DB command is not processed because an IMS Checkpoint is in progress.
X'00000010'	X'000041F4'	The UPDATE DB command is not processed because an MSDB Checkpoint is in progress.
X'00000010'	X'000041F8'	The UPDATE DB command is not processed because a takeover is in progress.
X'00000010'	X'00004200'	Commands are not processed because IMS shutdown is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004320'	The UPDATE DB command is not processed because another UPDATE DB START(QUIESCE) or UPDATE DB STOP(QUIESCE) command is in progress.
X'00000010'	X'00004325'	DB quiesce phase is in progress.
X'00000010'	X'00004400'	MINVERS in the RECON data sets is not 11.1.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005000'	The UPD DB command processing terminated because IMODULE GETSTOR storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.
X'00000014'	X'00005014'	The UPDATE DB command processing terminated because BCB storage could not be obtained.
X'00000014'	X'0000501C'	The UPD DB command processing terminated because IMODULE GETMAIN storage could not be obtained.
X'00000014'	X'00005FFF'	The UPDATE DB command processing terminated because of an internal error.

The following table includes an explanation of the completion codes. Errors unique to the processing of UPDATE DB command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 333. Completion codes for the UPDATE DB command

Completion code	Completion code text	Meaning
0		The command completed successfully for the resource.
8	COMMAND COMPLETE FOR SOME	Some. The command completed with error for some of the AREAS of the DEDB. Response lines for the area names in error are returned.
C	COMMAND COMPLETE FOR NONE	None. The command completed with error for all the AREAs of the DEDB. Response lines for the area names in error are returned. This error is returned when all the databases could not be quiesced.
10	NO RESOURCES FOUND	Database name is invalid, or the wildcard parameter specified does not match any database names.
17	ANOTHER CMD IN PROGRESS	This error is returned when the quiesce could not be started because another database command was in progress.
23	DB STOP ACCESS IN PROGRESS	A /DBRECOVERY, or UPDATE DB STOP(ACCESS) command to stop database access is in progress. This takes the database offline.
25	DB STOP UPDATES IN PROGRESS	A /DBDUMP or UPDATE DB STOP(UPDATES) command to stop database updates is in progress.
26	DEDB STOP IN PROGRESS	/DBRECOVERY, /STOP, or UPDATE DB STOP(SCHD) command to stop database scheduling is in progress for a DEDB.
31	NOT ALLOWED FOR A DEDB	Database is a DEDB. The command entered is not valid for the DEDB in the IMS environment.
32	NOT ALLOWED FOR AN MSDB	Database is an MSDB. The command entered is not valid for the MSDB in the IMS environment.
33	NOT ALLOWED FOR A HALDB MASTER	Command invalid HALDB master. The command OPTION is invalid for the HALDB master but partition structure rebuild will be done if structure rebuild is needed and if only one HALDB master was specified in the command. No rebuild will be attempted if there is more than one database name listed in the command. If there are multiple database names listed in the command and all are invalid except the HALDB master, then rebuild will be attempted if needed.

Table 333. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for an IMS descriptor or resource. DFSDSDB1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
53	NO RM ADDRESS SPACE	This error is returned when the command could not be processed because the RM address space is not present.
55	NO FAST PATH INSTALLED	No Fast Path installed. The command failed because Fast Path is not installed.
56	FF DB + LSO=Y + TRK = ERROR	Command is invalid on the RSR tracker because of the LSO=Y option.
65	DMB POOL STORAGE ERROR	DMB pool storage error. The command failed because of DMB pool storage request failure.
66	DMB POOL FULL	DMB pool full. The command failed because the DMB pool was full.
6C	NOT ALLOWED FOR A HALDB PARTITION	An UPDATE command specified a change to the residency option for a HALDB partition. The residency option is valid only for the master and not the partitions.
6F	REFERENCED BY PROGRAM	An UPDATE DB command is issued to change the resident option. There is a currently scheduled program that is referencing that database. The UPDATE command fails.
76	RECOVER CMD ACTIVE	/RECOVER START command is in progress to recover one or more databases with the database recovery services.
81	DBRC ERROR	DBRC error.
90	INTERNAL ERROR	Internal error. The command entered is not processed because of an internal error.
91	TIMEOUT ERROR	This error is returned when the quiesce could not be completed within the timeout period.
92	COMMAND PROCESSING ERROR	Command processing error. The command entered is not processed because of an error. A unique completion code could not be generated to explain the error. The message number and the return code that could not be converted to a completion code are listed in the error text.

Table 333. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
A0	DYNAMIC ALLOCATION FAILED	Dynamic allocation failed. The command entered has not completed processing because the dynamic allocation failed for the DB. Some of the command processing might be completed before the error is detected.
A1	DB IS AUTHORIZED BY BATCH	Database is authorized by batch. The command entered has not completed processing because the database is authorized by batch. Some of the command processing might be completed before the error is detected.
A2	DB IS AUTHORIZED BY ANOTHER IMS	Database is authorized by another IMS. The command entered has not completed processing because the database is authorized by another active or abnormally terminated IMS and its authorization state is incompatible with the current authorization request.
A3	AUTHORIZATION CHANGE FAILED	Authorization change failed. The DBRC CHNGAUTH request resulted in an error.
A4	DATABASE NOT REGISTERED TO DBRC	Database not registered to DBRC. The command processing has not completed because the database is not registered to DBRC. Some of the command processing might be completed before the error is detected.
A5	PREVENT FURTHER AUTH ON	Prevent further authorization ON. The command entered has not completed processing because the database or area is defined to DBRC as “prevent further authorization”. Some of the command processing might be completed before the error is detected.
A6	INVALID DATABASE RECORD IN RECON	Invalid database record in the RECON data set. The command entered has not completed processing because an invalid parameter was found during the evaluation process of the database usage compatibility. The database record might be invalid in the RECON data set. Some of the command processing might be completed before the error is detected.

Table 333. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
A7	DBRC UNAUTH FAILED FOR CHNGAUTH	DBRC unauthorization failed during change authorization. The command entered has not completed processing because of an error during UNAUTH processing for the change authorization request.
A8	INVALID DB RECORD IN RECON	Invalid database record in RECON. An UPDATE DB SET(ACCTYPE) command is entered to change the database authorization level. An encoded state of zero is returned by DBRC during the change authorization processing. Some of the command processing might be completed before the error is detected.
A9	DB OR AREA AUTHORIZATION ERROR	Database or area authorization error. For a database, the command entered has not completed processing because of a database authorization error. For an area, area authorization to DBRC failed. Some of the command processing might be completed before the error is detected.
AA	DB IN USE-BMP	The UPDATE DB command was not successful because the database is in use by a BMP. Some of the command processing might be completed before the error is detected.
AB	DB IN USE-DBCTL LONG THREAD	The UPDATE DB command was not successful because the database is in use by a long-running DBCTL thread. Some of the command processing might be completed before the error is detected.
AC	FP AREA HELD-LONG BUSY WAIT	The UPDATE DB command was not successful because the AREA of the DEDB is in long-busy wait. Some of the command processing might be completed before the error is detected.
AD	DYNAMIC UNALLOCATION FAILED	The UPDATE DB command was not successful due to a dynamic unallocation error. Some of the command processing might be completed before the error is detected.

Table 333. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
AE	DYNAMIC ALLOCATION ERROR	The UPDATE DB command was not successful due to a dynamic allocation error. No SVC99 is issued. Some of the command processing might be completed before the error is detected.
C1	OLR DDIR MISSING OR DFSPNT ZERO	Unknown DMB referenced for database. The command cannot be processed because an unknown data management block is referenced for the database. Refer to the DFS564I message put out to the system console to identify the DMB name that cannot be referenced.
CC	OLR IS ACTIVE FOR DATABASE	OLR is active for database. The command failed as OLR is active for the database.
D0	DATABASE CLOSE ERROR	Database close error. The command processing failed because of a database close error.
D1	DATABASE WRITE ERROR	Database write error. The command processing failed because of a database write error.
D2	DATABASE NEEDS BACKOUT	Database needs backout. The command processing failed as the database needs backout.
D3	DATABASE OR AREA NEEDS RECOVERY	Database or AREA needs recovery. The command processing failed as the database or area needs recovery.
D4	DATABASE NEEDS IMAGE COPY	Database needs image copy. The command processing failed as the database needs image copy.
D5	DATABASE HAS NO BACKOUTS	Database has no backouts. The command processing failed as there are no backouts for the database.
D6	DATABASE IN USE	Database in use. A SET(ACCTYPE) is specified for the DEDB and the authorization level cannot be changed as the DEDB is in use in a region.
D7	DB I/O PREVENTION NOT COMPLETE	Database I/O prevention not complete. The database cannot be started as it is extended because of an XRF takeover and the I/O prevention is not complete.
D8	DATABASE BACKOUTS PENDING	Database backouts pending. The access type specified for the database cannot be changed as restartable backouts are pending for the database.

Table 333. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
D9	DATABASE/AREA OPEN FAILED	Database open failed. The command failed because of an error opening the database. Refer to DFS0730I messages to determine the reason of the failure.
DA	DATABASE BEING RECALLED BY HSM	Database being recalled from HSM. The command processing failed because the database is being recalled from HSM.
DB	PARTITION OPEN FAILED	Partition open failed. The partitions open failed because the master is offline. This can also occur if the partition has been deleted and partition structure rebuild has occurred. Partition structure rebuild can be accomplished by issuing an UPD DB NAME(haldbmst) START(ACCESS) OPTION(OPEN) command, where haldbmst is the partition's master, or by issuing a qualified GU call for a key in the key range of the partition. List.recon can be used to determine if the partition exists or has been deleted.
DC	HALDB PARTITION BUILD FAILURE	Database partition build failure. The database partitions build for the DDIR or DMB failed. Refer to the DFS0415I message sent to the system console to determine the reason of the failure.
DD	HALDB PARTITION INIT FAILURE	Database partition initialization failed. The database partition initialization for the DDIR or DMB failed. Refer to the DFS0415 message sent to the system console for the details.
DE	ACBLIB READ FAILURE	ACBLIB read failure. The command is not processed because there was an error reading the ACBLIB.
DF	DB DIRECTORY INIT FAILURE	Database directory initialization failed. The command is not processed because of a database directory initialization failure.
E0	DATABASE OR AREA IN RECOVERY	Database or area in recovery. The command is not processed because the database or area is in recovery.

Table 333. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
E1	DB MUST BE STOPPED AND OFFLINE	The database must be stopped and taken offline in order for the attribute to be changed. You might need to issue a /DBR DB command or an UPDATE DB STOP(ACCESS) command to stop the database and take it offline before issuing the UPDATE DB SET(RESIDENT(Y|N)) command.
E2	PARALLEL DB OPEN NOT COMPLETE	Restart parallel DB open not complete. The command is not processed because the restart parallel DB open is not complete for the database.
E5	PARTICIPANT UNABLE TO QUIESCE	This error is returned when the quiesce could not be completed successfully across the IMSplex. The IMS with this completion code was the quiesce participant that was not able to be quiesced.
E6	QUIESCE COMMUNICATION FAILURE	This error is returned when the quiesce could not be completed because of a failure to communicate across the IMSplex. There could be a problem with RM, OM, or SCI that has caused the communication failure to occur.
E7	CMD NOT ALLOWED	This error is returned when the database command could not be processed because a quiesce command was in progress.
E8	DATABASE HAS INTENT TO REORGANIZE	This error is returned when the quiesce could not be started because the RECON data sets indicate that there is an intent to reorganize the database.
E9	DB IN WRONG STATE TO BE QUIESCED	This error is returned when the named resource is in the wrong state for quiesce processing to proceed.
EE	DATABASE BACKOUT ERROR	Database backout error. The command processing failed because of a database backout error.
EF	DATABASE IS IN ERROR	Database is in error. The command entered is not processed because the database is in error.
F0	NO AREA LOCK	No AREA lock. The command processing failed as the area lock could not be obtained.
F1	AREA NOT STOPPED	Area not stopped. The command entered is not processed because the AREA is not stopped.

Table 333. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
F2	PRELOAD IS ACTIVE FOR AREA	Preload is active for AREA. The command entered is not processed because the AREA is not stopped.
F3	UNRESOLVED INDOUBTS FOR AREA	Unresolved indoubts for AREA. The command entered is not processed because unresolved indoubts exist for the AREA.
F4	ALLOCATION FAILED	Allocation failed. Allocation failed for the AREA name.
F5	AREA NEEDS RECOVERY	AREA needs recovery. The command processing failed because the area needs recovery.
F6	ADS NUMBER DISCREPANCY	ADS number discrepancy. The command failed for the AREA because there is a discrepancy between the number of ADS allocated by IMS and the number of ADS known to DBRC. Correct the discrepancy and reissue the command to deallocate the data sets.
F7	AREA IS NOT LOADED INTO CF	Area is not loaded into CF. The AREA OPEN failed and is not loaded into the Coupling Facility.
F8	AREA HAS I/O TOLERATED CI	Area has I/O tolerated CI. The command is not processed because the AREA has an I/O tolerated CI.
F9	AREA HAS 2ND CI EEQE	Area has 2nd CI EEQE. The command is not processed because the AREA has a 2nd CI EEQE.
FC	UTILITY ACTIVE ON AREA	This error is returned when the area is in use by a utility.
FD	AREA HAS EEQE	This error is returned when the area has an extended error queue element (EEQE).
FE	AREA HAS EQE	This error is returned when the area has an error queue element (EQE).
14A	NOT-REACHABLE	Command failed for the IMS because its SCI was not active. When SCI is not active, the IMS is not reachable.
14B	SHUTDOWN	Command failed for the IMS because it shut down normally during database quiesce processing.
14C	ABENDED	Command failed for the IMS because it terminated abnormally during database quiesce processing.

Examples

The following are examples of the UPDATE DB command:

Example 1 for UPDATE DB command

TSO SPOC input:

```
UPDATE DB NAME(DEDBJ00%,BADNAME,BAD*) SET(RESIDENT(Y))
```

TSO SPOC output:

```
Response for: UPDATE DB NAME(DEDBJ00%,BADNAME,BAD*) SET(RESIDENT(Y))
DBName   MbrName    CC CCText
BAD*     IMS1       10 NO RESOURCES FOUND
BADNAME  IMS1       10 NO RESOURCES FOUND
DEDBJ001 IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ002 IMS1        0
DEDBJ003 IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ004 IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ005 IMS1        0
DEDBJ006 IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ007 IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ008 IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ009 IMS1       E1 DATABASE HAS NOT BEEN DBR"D
```

OM API input:

```
CMD(UPDATE DB NAME(DEDBJ00%,BADNAME,BAD*) SET(RESIDENT(Y)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.311 00:34:47.028100</statime>
<stotime>2006.311 00:34:47.031559</stotime>
<staseq>BFAADA74E4584E87</staseq>
<stoseq>BFAADA74E5307115</stoseq>
<rqsttkn1>USRT011 10163446</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>DB              </kwd>
<input>UPDATE DB NAME(DEDBJ00%,BADNAME,BAD*) SET(RESIDENT(Y)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
```

```
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="GBL" llbl="Global" scope="GBL" sort="d" key="2" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="y" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(DEDBJ001) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(BADNAME ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>DB(BAD*    ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp
<rsp>DB(DEDBJ002) MBR(IMS1) CC(   0) </rsp>
<rsp>DB(DEDBJ003) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ004) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ005) MBR(IMS1) CC(   0) </rsp>
<rsp>DB(DEDBJ006) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ007) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ008) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ009) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update some databases to be resident. The update succeeds for databases DEDBJ002 and DEDBJ005, as shown by completion code 0. The update fails for several databases with completion code E1, because the databases need to be taken offline by use of the /DBRECOVERY command in order to change a runtime resource definition attribute. The update fails for database BADNAME and for parameter BAD* with completion code 10, since database BADNAME does not exist and no database name starts with BAD.

Example 2 for UPDATE DB command

TSO SPOC input:

```
UPDATE DB NAME(DEDBJN22) AREA(*) START(ACCESS) OPTION(OPEN)
```

TSO SPOC output:

```
DBName    MbrName      CC    CCtext
DEDBJN22  IMS1            C
DB22AR0   IMS1            0
DB22AR1   IMS1            A9   DB OR AREA AUTHORIZATION ERROR
```

Explanation: In this example, CC=A9 means that DBRC returned a nonzero return code for authorization request.

Example 3 for UPDATE DB command

TSO SPOC input:

```
UPD DB NAME(DEDBJN22) START(QUIESCE) OPTION(HOLD) SET(TIMEOUT(60))
```

TSO SPOC output:

```
DBName    MbrName  CC
DEDBJN22  IMS1      0
DEDBJN22  IMS2      0
DEDBJN22  IMS3      0
```

Explanation: This example is of a successful quiesce and hold for Fast Path DEDB DEDBJN22, which is coordinated across three IMS systems.

Example 4 for UPDATE DB command

TSO SPOC input:

```
UPD DB NAME(DEDBJN22) STOP(QUIESCE)
```

TSO SPOC output:

```
DBName   AreaName MbrName  CC CCText
DEDBJN22          IMS1      C COMMAND COMPLETE FOR NONE
DEDBJN22 DB22AR0  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR1  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR2  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR3  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR4  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
```

Explanation: This example attempts to release a quiesce on a DEDB. DEDB DEDBJN22 consists of five areas. The command failed because none of the areas were held in quiesced state. A response line is returned for the DEDB name with CC=C ("COMMAND COMPLETE FOR NONE") because none of the areas under the DEDB were in a state to be released. A response line is also returned for each area.

Example 5 for UPDATE DB command

TSO SPOC input:

```
UPD DB NAME(DEDBJN22) STOP(QUIESCE)
```

TSO SPOC output:

```
DBName   AreaName MbrName  CC CCText
DEDBJN22          IMS1      8 COMMAND COMPLETE FOR SOME
DEDBJN22 DB22AR0  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR3  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR4  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
```

Explanation: This example attempts to release a quiesce on a DEDB. DEDB DEDBJN22 consists of five areas. The command was partially successful because it was able to release the quiesce on DB22AR1 and DB22AR2 only. In this example, DB22AR0, DB22AR3, and DB22AR4 were not in quiesced state. A response line is returned for the DEDB name with CC=8 ("COMMAND COMPLETE FOR SOME") because only some of the areas under the DEDB were in a state to be released. A response line is also returned for each area that could not be processed.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related tasks:

Updating runtime database resource and descriptor definitions with the UPDATE command (System Definition)

Related reference:

/LOCK command (Commands)

Command keywords and their synonyms (Commands)

"/START DB command" on page 465

"/STOP DB command" on page 516

"/UNLOCK DB command" on page 620

UPDATE DBDESC command

Use the UPDATE DBDESC command to update database descriptors. A descriptor is a model that can be used to create descriptors or resources.

Updating a descriptor changes only the attributes explicitly specified on the UPDATE command. Attributes not specified retain their existing values. Any database resource or descriptor can be created using this descriptor as a model, by specifying the CREATE command with LIKE(DESC(*descriptor_name*)). Any descriptor or resource that was already created using this descriptor is not updated.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 687
- "Usage notes" on page 688
- "Output fields" on page 689
- "Return, reason, and completion codes" on page 689
- "Examples" on page 691

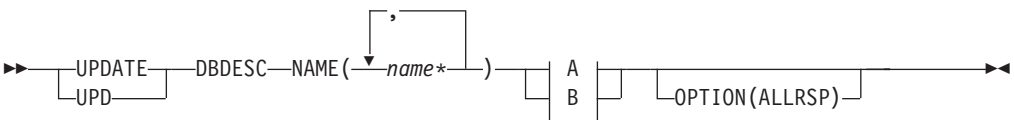
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

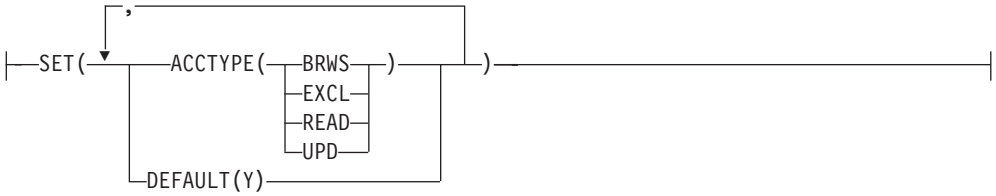
Table 334. Valid environments for the UPDATE DBDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE DBDESC	X	X	
NAME	X	X	
OPTION	X	X	
SET	X	X	

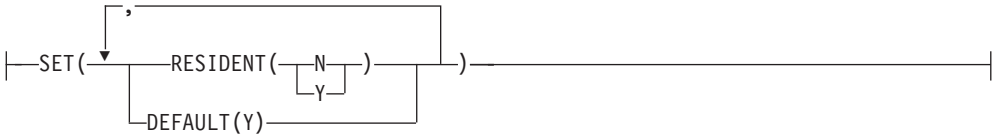
Syntax



A:



B:



Keywords

The following keywords are valid for the UPDATE DBDESC command:

NAME
Specifies the 1-8 character name of the database descriptor name. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the descriptor names that are processed. For NAME(*) command responses are returned only for the descriptor names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the descriptor names that are processed.

OPTION
Specifies additional functions to be performed along with the command.

ALLRSP
Indicates that the response lines are to be returned for all descriptors that are processed on the command. The default action is to return response lines only for the descriptors that resulted in an error. It is only valid with names that include a wildcard character. ALLRSP is ignored for other NAME values.

SET
Specifies the attribute values to be changed.

ACCTYPE
Specifies the access intent for a database created using this descriptor.

BRWS
The database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data since a sharing IMS subsystem could be updating the database. The database is opened for read-only processing.

EXCL
The database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC.

READ
The database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. Access type READ differs from access type BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UPD
The database is for update as well as read processing in the IMS subsystem.

DEFAULT(Y)
Specifies this descriptor as the default, which resets the existing default descriptor to DEFAULT(N). When a descriptor is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a database descriptor called DFSDSDB1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor. Since only one database descriptor can be the default at one time, only one database name can be specified with DEFAULT(Y).

RESIDENT
Specifies the resident option of a database created using this descriptor. The RESIDENT(Y) option takes effect at the next restart, unless the database was updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart. A database defined as a DEDB in ACBLIB always sets the RESIDENT(Y) attribute when the DEDB is loaded, regardless of the RESIDENT value specified. RESIDENT(N) is rejected for a DEDB.

N The DMB associated with the named database resource is not made resident in storage. The DMB is loaded at scheduling time.

Y The DMB associated with the named database resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the DMB and initializes it. A resident database is accessed from local storage, which eliminates I/O to the ACBLIB. In an online environment, the DMB control blocks are stored in the ACBLIB. If the DLI/SAS address space exists, DLI/SAS loads the DMB, otherwise, it is the IMS control region that loads it. This makes the DMB dependent on the existence of the corresponding database resource.

Usage notes

Descriptors exist for the life of the IMS unless they are deleted using a DELETE command. Descriptors are recoverable across an IMS warm start or emergency restart. Descriptors are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

If all the attributes specified by the UPDATE command are already defined for the descriptor, no update is actually made, no descriptors are quiesced, no log record

is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

The UPDATE DBDESC command can be issued only through the OM API. This command applies to DB/DC and DBCTL systems. The UPDATE DBDESC commands are not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE DBDESC commands are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined). This command is recoverable.

Each descriptor is updated individually. Individual updating does not work like online change where either all descriptors are updated or no descriptors are updated. Descriptors can be successfully updated if they are not currently in use. If a descriptor is in use, the update fails. In a sysplex environment, the update might succeed on some IMS systems and fail on others. A descriptor is in use if another command is in progress that references the command.

If the descriptor is the IMS-defined database descriptor (DFSDSDB1), the only attribute that can be updated is the DEFAULT attribute.

Output fields

The following table shows the UPDATE DBDESC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 335. Output fields for the UPDATE DBDESC command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the non-zero completion code.
DESC	DBDESC	Database descriptor name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a non-zero completion code and contains information that further explains the completion code.
OLDDEF	DBDESC	Old default descriptor name, if this descriptor is updated to be the default by specifying DEFAULT(Y). The old default descriptor is no longer the default.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 336. Return and reason codes for the UPDATE DBDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000004'	X'00002008'	Invalid number of keywords. Either a SET keyword is required.
X'00000008'	X'00002048'	Invalid SET attribute.
X'00000008'	X'00002133'	Multiple name parameters were specified with DEFAULT(Y). Only one descriptor can be the default at a time.
X'0000000C'	X'00003000'	Command was successful for some descriptors but failed for others. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), output lines are only returned for descriptors with non-zero completion codes. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004310'	Storage could not be obtained for the Transaction Input edit routine table. A cold start is required to fix this error.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 337. Completion codes for the UPDATE DBDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for database resource or database resource descriptor.
17	ANOTHER CMD IN PROGRESS	None. The command completed with error for all the Areas of the DEDB. Response lines for the area names in error are returned.

Table 337. Completion codes for the UPDATE DBDESC command (continued)

Completion code	Completion code text	Meaning
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for an IMS descriptor or resource. DFSDSDB1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
6C	NOT ALLOWED FOR A HALDB PARTITION	The UPDATE command specified a change to the residency option for a HALDB partition. The residency option is valid only for the master and not the partitions.
8A	WILDCARD PARAMETER NOT SUPPORTED	A wildcard parameter was specified with DEFAULT(Y). Only one descriptor can be the default at a time.

Examples

The following are examples of the UPDATE DBDESC command:

Example 1 for UPDATE DBDESC command

TSO SPOC input:

```
UPDATE DBDESC NAME(*) SET(RESIDENT(Y)) OPTION(ALLRSP)
```

TSO SPOC output:

```
Response for: UPDATE DBDESC NAME(*) SET(RESIDENT(Y)) OPTION(ALLRSP)
DescName MbrName    CC CCText
BRWSDESC IMS1        0
DESC001  IMS1        0
DESC002  IMS1        0
DESC003  IMS1        0
DESC004  IMS1        0
DESC005  IMS1        0
DFSDSDB1 IMS1       48 NOT ALLOWED FOR IMS RESOURCE
EXCLDESC IMS1        0
RESDESC  IMS1        0
```

OM API input:

```
CMD(UPDATE DBDESC NAME(*) SET(RESIDENT(Y)) OPTION(ALLRSP))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.312 23:27:37.656606</statime>
<stotime>2006.312 23:27:37.661922</stotime>
<staseq>BFAD4F2D1B11EC8E</staseq>
<stoseq>BFAD4F2D1C5E294E</stoseq>
<rqsttkn1>USRT011 10152737</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
```

```
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>DBDESC          </kwd>
<input>UPDATE DBDESC NAME(*) SET(RESIDENT(Y)) OPTION(ALLRSP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1" scroll=
 "no" len="8" dtype="CHAR" align="left" skipb="no"  />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(EXCLDESC) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC004 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC005 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC001 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(RESDESC ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(BRWSDESC) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC002 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC003 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DFSDSDB1) MBR(IMS1) CC(  48) CCTXT(NOT ALLOWED FOR IMS
RESOURCE) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update all database descriptors to be resident. The update completed successfully for most database descriptors, as shown by completion code 0. The update failed for IMS-defined descriptor DFSDSDB1 with completion code 48 `NOT ALLOWED FOR IMS RESOURCE`, since the only attribute that can be updated for DFSDSDB1 is DEFAULT(Y).

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

UPDATE IMS command

Use the UPDATE IMS command to update local and global IMS attributes, which include global status for areas, databases, and transactions in an IMSplex.

To change global PLEXPARM values in an IMSplex, use the UPDATE IMS SET(PLEXPARM()) command. To change local LCLPARM values in one or more IMS systems in an IMSplex, use the UPDATE IMS SET(LCLPARM()) command.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 696
- "Output fields" on page 697
- "Return, reason, and completion codes" on page 697
- "Examples" on page 699

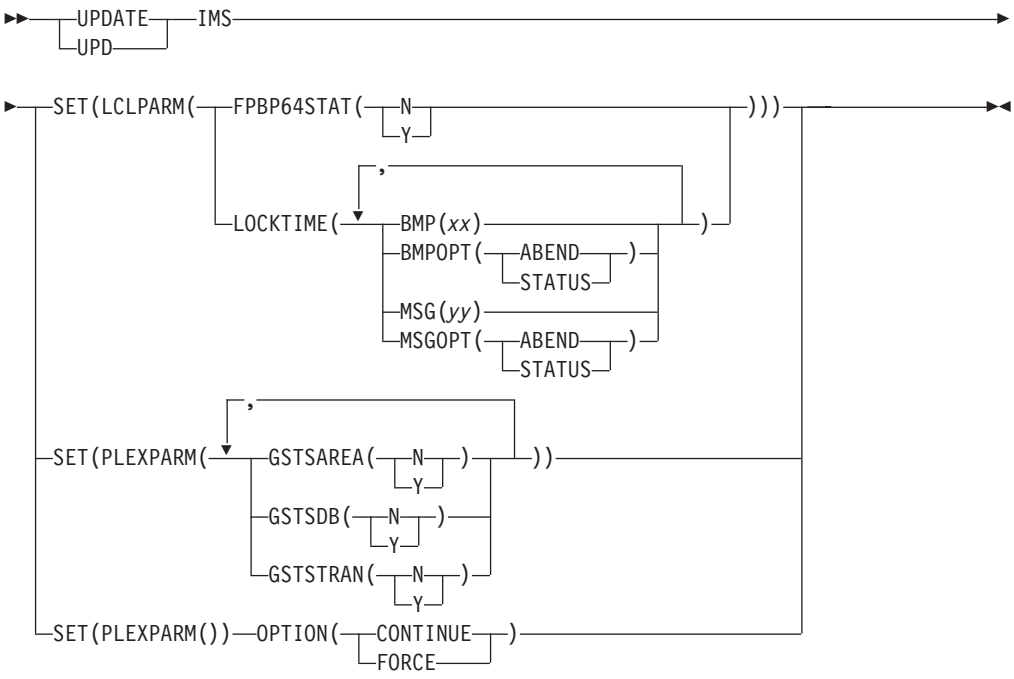
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE IMS command and keywords can be issued.

Table 338. Valid environments for the UPDATE IMS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE IMS	X	X	X
SET	X	X	X
OPTION	X	X	X

Syntax



Keywords

The following keywords are valid for the UPDATE IMS command:

LCLPARM()
Specifies the values for the local parameters to be modified. The LCLPARM() keyword is mutually exclusive with the PLEXPARM() keyword in the UPDATE IMS SET() command.

The UPDATE IMS SET(LCLPARM()) command can be used to modify local parameters such as FPBP64STAT at each IMS system.

For the UPDATE IMS SET(LCLPARM()) command, all IMS systems that received the command from OM process the command.

Specify one of the following options:

FPBP64STAT
Specifies whether Fast Path 64-bit buffer usage statistics per unit of work (UOW) for dependent regions are to be written to OLDS. The Fast Path 64-bit buffer usage is recorded in X'5945' log records.

N Does not write Fast Path 64-bit buffer usage statistics per unit of work for dependent regions in X'5945' log records to OLDS.

Y Writes Fast Path 64-bit buffer usage statistics per unit of work for dependent regions in X'5945' log records to OLDS.

LOCKTIME
Specifies the IMS LOCKTIME values. You can specify LOCKTIME values in any combination of the following keywords and parameters:

BMP(*xx*)
Specifies the amount of time that IMS waits before lock requests for BMP regions are timed out. BMP regions include IMS BMP and JBP regions. The value, which represents time in seconds, can range 1 - 32767.

BMPOPT(ABEND | STATUS)
Specifies whether IMS ends a timed-out task abnormally (ABEND) or returns a status code to the application (STATUS).

MSG(*yy*)
Specifies the amount of time that IMS waits before lock requests for MSG regions are timed out. MSG regions include IMS MPP, JMP, and IFP regions, as well as DRA threads. The value, which represents time in seconds, can range 1 - 32767.

MSGOPT(ABEND | STATUS)
Specifies whether IMS ends a timed-out task abnormally (ABEND) or returns a status code to the application (STATUS).

Updating IMS LOCKTIME values does not affect the lock timeout value in an IRLM. To change the lock timeout value in an IRLM, use the existing MODIFY IRLM command.

Recommendation: Generally, use the same timeout values for both IMS and IRLM. When using two values in IMS, setting the IRLM timeout value to the lower of the two IMS values allows IMS and IRLM to act together. When more than one IMS is identified to the same IRLM, workload conditions might require using different timeout values across IMS systems. In such a case, note the following:

- Using a lower timeout value in IRLM results in lock requests to wait until the time spent for waiting equals the lowest IMS LOCKTIME value.

- Using a higher timeout value in IRLM results in lock requests to wait beyond the IMS time because IRLM does not call IMS until the IRLM timeout value is exceeded.

PLEXPARM()
Specifies the values for the global parameters to be modified. The LCLPARM keyword is mutually exclusive with the PLEXPARM keyword in the UPDATE IMS SET() command.

The UPDATE IMS SET(PLEXPARM()) command can be used to modify global status for areas, databases, and transactions that is maintained in an IMSplex. If the command is successful, the values in the RM global PLEXPARM entry along with the values maintained in each IMS system are updated.

For the UPDATE IMS SET(PLEXPARM()) command, all IMS systems in an IMSplex receive the command, but only the command master IMS processes the command. The non-master IMS systems do not process the command that is sent from OM. Instead, the command master IMS coordinates the change across all IMS systems in the IMSplex.

Specify one or more of the following resources:

GSTSAREA
Specifies how area status is maintained in the IMSplex.

N No global status is maintained for area resources in RM. All global status in RM for area resources is deleted.

Y Global status is maintained for area resources in RM. Global status is maintained on subsequent UPD AREA SCOPE(ALL) commands or type-1 area commands that include a GLOBAL keyword.

GSTSDB
Specifies how database status is maintained in the IMSplex.

N No global status is maintained for database resources in RM. All global status in RM for database resources is deleted.

Y Global status is maintained for database resources in RM. Global status is maintained on subsequent UPD DB SCOPE(ALL) commands or type-1 database commands that include a GLOBAL keyword.

GSTSTRAN
Specifies how transaction status is maintained in the IMSplex.

N No global status is maintained for transaction resources in RM. All global status in RM for transaction resources is deleted.

Y Global status is maintained for transaction resources in RM. Global status is maintained on subsequent UPD TRAN SCOPE(ALL) commands.

OPTION()
Specifies options for the UPDATE IMS SET command.

CONTINUE
Indicates that IMS should continue to process a previously entered UPDATE IMS SET command that encountered an error before it completed. The command must be at a point that allows processing to continue, otherwise the command is aborted.

If OPTION(CONTINUE) is specified, no parameters can be passed on the PLEXPARM parameter.

FORCE
OPTION(FORCE) can be used in a DBCTL warm-standby environment when the UPD IMS command fails because the standby did not finish restart processing. If OPTION(FORCE) is specified, the UPD IMS command can be completed even when an IMS is in restart mode. When OPTION(FORCE) is specified, the IMS in restart mode does not participate and update the UPD IMS values; this option only allows the UPD IMS command to finish at other IMS systems. The DBCTL standby rereads the global entry when it restarts to become the active in order to obtain the current PLEXPARM values. If OPTION(FORCE) is used when a non-DBCTL standby system did not finish restart, the results might not be as expected because the IMS does not run the UPD IMS command. This could lead to errors because of mismatch in the PLEXPARM values.

Usage notes

The UPDATE IMS SET command is a type-2 command that must be issued through the OM API. The command is defined to OM as ROUTE=ALL. OM routes the command to all IMS systems in an IMSplex. This command is not a recoverable command.

In general, when a nonzero return code is received for the UPDATE IMS command, you must enter the UPD IMS SET(PLEXPARM()) OPTION(CONTINUE) command, which tells IMS to either complete the command or to cancel the command and clean up as required. The reason code from the UPD IMS SET(PLEXPARM()) OPTION(CONTINUE) command might indicate the state of the original UPD IMS command. If the command is unable to determine the state of the original UPD IMS command, the reason code indicates that fact. The user can use the QUERY IMS command to find out the state of the global IMSplex parameters.

If an error is encountered during the UPD IMS command processing, another new UPD IMS command cannot be processed until the UPDATE IMS SET(PLEXPARM()) OPTION(CONTINUE) command is issued and completes successfully.

If the error condition that caused the nonzero return code to be received for the UPDATE IMS command is not resolved, the UPDATE IMS SET(PLEXPARM()) OPTION(CONTINUE) command continues to encounter the same error condition. The error condition must be resolved before the command can be cleaned up and another UPDATE IMS command can be entered.

This command is not valid on an RSR tracker, an FDBR system, or an XRF alternate system. However, an FDBR system and an XRF alternate system maintain global PLEXPARM values internally and report their values with a QUERY IMS SHOW (PLEXPARM) command. IMS systems on RST tracker or FDBR systems return a completion code of ICC_NA or 1. The UPD IMS command is allowed to complete at other IMS systems.

The UPD IMS SET command is not allowed if an IMS is in restart. The command fails with a 'B0' return code. The UPD IMS command must be issued after the IMS completed restart processing.

Output fields

The following table shows the UPDATE IMS output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 339. Output fields for the UPDATE IMS SET command

Short label	Keyword	Scope	Meaning
CC	n/a	n/a	The completion code for the line of output. The completion code is always returned.
LPARM	LCLPARM	LCL	The name of the local parameter that is being changed. Multiple subparameters of this parameter might be changed depending on what was entered in the command.
MBR	n/a	n/a	The IMS identifier of the IMS that built the output line. The IMS identifier is always returned.
CCTXT	n/a	LCL	Completion code text returned to provide more information about the completion code. CC text could include a return code from a service. CC text is returned only if the completion code is nonzero.
PARMNAME	PARMNAME	LCL	The name of the global parameter that is being changed. Multiple sub-parameters of this parameter may be changed depending on what was entered in the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMS command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 340. Return and reason codes for the UPDATE IMS command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMS command completed successfully.
X'00000004'	X'00001000'	The UPDATE IMS command was not processed on the IMS system because the IMS system is not the command master. No resource information is returned.

Table 340. Return and reason codes for the UPDATE IMS command (continued)

Return code	Reason code	Meaning
X'00000004'	X'00001020'	UPD IMS OPTION(CONTINUE) finished cleanup but state is not known. User must redo or reissue the command.
X'00000008'	X'00002004'	Both LCLPARM and PLEXPARM keywords are specified on the UPDATE IMS SET() command. LCLPARM and PLEXPARM are mutually exclusive keywords.
X'00000008'	X'00002008'	Neither the PLEXPARM keyword nor the LCLPARM keyword is specified on the UPDATE IMS SET() command. Specify PLEXPARM for global parameters or LCLPARM for local parameters.
X'00000008'	X'00002040'	More than one filter value is specified on the UPDATE IMS command.
X'00000010'	X'00004000'	The UPDATE IMS command is not processed as IMS restart is not complete.
X'00000010'	X'00004004'	The UPDATE IMS command is not processed as CQS is not available.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	The UPDATE IMS command is not processed as the resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'00004100'	The UPDATE IMS command is not processed as the resource structure is full.
X'00000010'	X'00004104'	The UPDATE IMS command is not processed as RM is not available.
X'00000010'	X'00004108'	The UPDATE IMS command is not processed as SCI is not available.
X'00000010'	X'00004124'	The UPDATE IMS command is not processed as a process step is in progress.
X'00000010'	X'00004128'	The UPDATE IMS command is not processed as no RM process step is in progress.
X'00000010'	X'00004200'	The UPDATE IMS command is not processed as IMS SHUTDOWN is in progress.
X'00000010'	X'00004208'	Invalid command for RMENV=NO.
X'00000010'	X'0000420C'	Error on non-command master. The UPD IMS could not complete because of an error on the non master IMS.
X'00000014'	X'00005000'	The UPDATE IMS command is not processed as the IMODULE GETSTOR storage could not be obtained.
X'00000014'	X'00005004'	The UPDATE IMS command processing terminated because a DFSOCMD response buffer could not be obtained.

Table 340. Return and reason codes for the UPDATE IMS command (continued)

Return code	Reason code	Meaning
X'00000014'	X'00005100'	The UPDATE IMS command is not processed because of a RM error.
X'00000014'	X'00005104'	The UPDATE IMS command is not processed because of a CQS error.
X'00000014'	X'00005108'	The UPDATE IMS command is not processed because of a SCI error.
X'00000014'	X'0000510C'	The UPDATE IMS command is not processed as another process is in progress.
X'00000014'	X'00005FFF'	The UPDATE IMS command is not processed because of an IMS internal error.

Errors that are unique to the processing of the UPDATE IMS command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 341. Completion codes for the UPDATE IMS command

Completion code	Completion code text	Meaning
0		The UPDATE IMS command completed successfully.
1		Command is not applicable to the IMS.
55	NO FASTPATH INSTALLED	Command is not applicable because Fast Path is not installed.
B0		IMS restart is not complete. The UPD IMS command could not complete because an IMS did not complete restart. You need to terminate the UPD IMS command with an UPD IMS SET(PLEXPARM()) OPTION(CONTINUE). The UPD IMS SET(PLEXPARM(...)) command must be issued after the IMS finishes restart processing.
BB		Command is not applicable as RMENV=NO.
14A	NOT-REACHABLE	Command failed for the IMS because its SCI was not active. When SCI is not active, the IMS is not reachable.
14B	SHUTDOWN	Command failed for the IMS because it shut down normally during command processing.
14C	ABENDED	Command failed for the IMS because it terminated abnormally during command processing.
146	INVALID KEYWORD FOR DCCTL	Command is not applicable in a DCCTL system.
147	FPBP64 NOT ENABLED	Command is not applicable because Fast Path 64-bit buffer manager is not enabled.

Examples

The following are examples of the UPDATE IMS command:

Example 1 for UPDATE IMS command

TSO SPOC input:

```
UPDATE IMS SET(PLEXPARM(GSTSAREA(Y),GSTSDB(Y),GSTSTRAN(N)))
```

TSO SPOC output:

```
Response for UPDATE IMS SET(PLEXPARM(GSTSAREA(Y),GSTSDB(Y),GSTSTRAN(N)))
ParmName  MbrName    CC
PLEXPARM  PRODIMS1    0
PLEXPARM  PRODIMS2    0
PLEXPARM  SYS3        0
```

Explanation: The UPD IMS command is processed successfully at the three active IMS systems (PRODIMS1, PRODIMS2, and SYS3) in the IMSplex and is updated successfully in RM.

Example 2 for UPDATE IMS command

TSO SPOC input:

```
UPDATE IMS SET(PLEXPARM(GSTSTRAN(Y)))
```

TSO SPOC output:

```
Response for UPDATE IMS SET(PLEXPARM(GSTSTRAN(Y)))
ParmName  MbrName  CC       GSTSTRAN
PLEXNAME  PRODIMS1  0       Y
PLEXNAME  PRODIMS2  0       Y
PLEXNAME  SYS3      0       Y
```

Explanation: The UPD IMS command is processed successfully at the three active IMS systems (PRODIMS1, PRODIMS2, and SYS3) in the IMSplex and is updated successfully in RM.

Example 3 for UPDATE IMS command

TSO SPOC input:

```
UPDATE IMS SET(PLEXPARM(GSTSTRAN(N)))
```

TSO SPOC output:

```
Response for: UPDATE IMS SET(GSTSTRAN(Y))
MbrName   CC   GSTSTRAN CCText
SYS3       0   Y
PRODIMS1   0   Y
PRODIMS2  16   Y        IMS TIMED OUT
```

Example 4 for UPDATE IMS command

TSO SPOC input:

```
UPDATE IMS SET(LCLPARM(FPBP64STAT(Y)))
```

TSO SPOC output:

```
    ParmName LclParmName MbrName    CC
    LCLPARM  FPBP64STAT  IMS1        0
    LCLPARM  FPBP64STAT  IMS2        0
```

Explanation: In this example, the UPDATE IMS command turns on FPBP64STAT logging.

Example 5 for UPDATE IMS command

TSO SPOC input:

```
UPDATE IMS SET(LCLPARM(FPBP64STAT(N)))
```

TSO SPOC output:

```
    ParmName LclParmName MbrName    CC
    LCLPARM  FPBP64STAT  IMS1        0
    LCLPARM  FPBP64STAT  IMS2        0
```

Explanation: In this example, the UPDATE IMS command turns off FPBP64STAT logging.

Example 6 for UPDATE IMS command

TSO SPOC input:

```
UPD IMS SET(LCLPARM(LOCKTIME(MSG(40),MSGOPT(ABEND),BMP(30),
BMPOPT(STATUS))))
```

TSO SPOC output:

```
  ParmName LclParmName MbrName    CC
  LCLPARM  LOCKTIME    IMS1        0
  LCLPARM  LOCKTIME    IMS2        0
```

Explanation: In this example, the UPDATE IMS command updates IMS LOCKTIME values.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

DFSDFxxx member of the IMS PROCLIB data set (System Definition)

UPDATE LE command

Use the UPDATE LE command to define Language Environment (LE) runtime parameter overrides or to change the system option to enable or disable LE override processing.

The parameters can be filtered by a transaction code, LTERM name, user ID, or program name for MPP and JMP regions. The parameters may be filtered by a program name for IFB, BMP, and JBP regions. Message driven BMP regions can also filter on a transaction code. Any combination of parameters may be used to qualify the application instance to which the runtime parameters are applied. The first available entry in the table is used. The new entry may be added before or after existing entries, depending on where free space exists in the table.

Subsections:

- "Environment" on page 702
- "Syntax" on page 702
- "Keywords" on page 702
- "Usage notes" on page 703
- "Output fields" on page 703
- "Return, reason, and completion codes" on page 704

• "Examples" on page 705

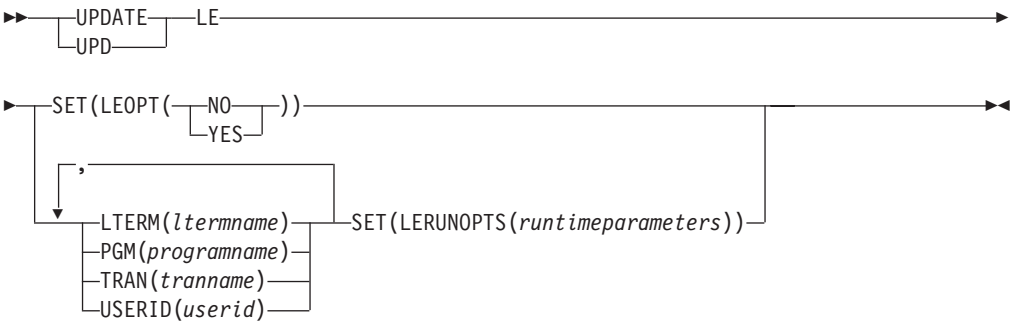
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE LE command and keywords can be issued.

Table 342. Valid environments for the UPDATE LE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE LE	X	X	X
LTERM	X	X	X
PGM	X	X	X
SET	X	X	X
TRAN	X	X	X
USERID	X	X	X

Syntax



Keywords

The following keywords are valid for the UPDATE LE command:

LTERM()
Specifies the 1-8 character name of the lterm to which the parameters are to be defined.

PGM()
Specifies the 1-8 character name of the program to which the parameters are to be defined.

SET() Specifies the updates to attributes or parameters.

LEOPT()
Specifies whether LE runtime parameters can be overridden dynamically for all active IMS systems in the IMSplex. YES indicates that the IMS systems enable overrides which enables the DL/I INQY call to retrieve runtime parameters. NO indicates that the IMS systems do not enable dynamic overrides to the parameters.

This option allows the user to override the LEOPT=Y|N option that is specified in the DFSCGxxx PROCLIB member without having to bring down the IMS system.

When runtime overrides are disabled (IMS is started with LEOPT=N or the UPD LE SET(LEOPT(NO)) command is issued) the runtime parameter table continues to be updated as UPD LE SET(LERUNOPTS()) or DEL LE commands are issued. If the UPD LE SET(LEOPT(YES)) command is then specified, all changes that were made during the time overrides were disabled are available.

This command is automatically routed to all IMS systems that are active in the IMSplex. The user can not override the OM routing to route to a single IMS.

LERUNOPTS()
Specifies the LE dynamic runtime parameters. If an existing set of parameters is found for the specified TRAN, LTERM, USERID, or PGM, the new parameter string completely replaces the existing parameter string. The parameters are not appended to the existing string. The string is only replaced when the specified filters are an exact match for the existing entry.

If there is no existing entry, then the first unused entry that is large enough to contain the parameters is used for the update. An unused entry is one that was previously deleted by the DEL LE command. If there are no unused entries or none that are large enough, storage is allocated for a new entry and the entry is added to the top of the table.

The UPDATE LE SET(LERUNOPTS()) command is processed regardless of the LEOPT system option. This means that IMS continues to build the runtime parameter table entries even though they will not be retrieved through the DL/I INQY call.

This command is automatically routed to all IMS systems that are active in the IMSplex. The user cannot override the parameters on a single IMS.

TRAN()
Specifies the 1-8 character name of the transaction to which the parameters are to be defined.

USERID()
Specifies the 1-8 character name of the user ID to which the parameters are to be defined.

Usage notes

This command may be specified only through the Operations Manager API. The command syntax for this command is defined in XML and is available to automation programs that communicate with OM.

OM overrides the routing on the command and routes the command to all IMS systems in the IMSplex. The user specified route list is ignored.

At least one of the resource filters (TRAN, LTERM, USERID, or PGM) must be specified.

Output fields

The following table shows the UPDATE LE output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. ERR appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 343. Output fields for the UPDATE LE command

Short label	Keyword	Meaning
CC	N/A	Completion code for the line of output. Completion code is always returned.
MBR	N/A	IMSplex member (IMS identifier) that built the output line. Member name is always returned.

Return, reason, and completion codes

The OM return and reason codes that may be returned as a result of this command are standard for all commands entered through the OM API. An IMS return and reason code is returned to OM by the UPDATE LE command.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE LE command completed successfully.
X'00000008'	X'00002008'	No keywords were specified on the command. At least one keyword is required. When SET(LERUNOPTS()) is specified, at least one resource name must be specified.
X'00000008'	X'00002010'	An asterisk or percentage sign was specified in the filter name. Wildcards are not supported on the command.
X'00000008'	X'00002014'	An invalid character was specified in the filter name.
X'00000008'	X'00002040'	An invalid filter was specified on the command. When SET(LEOPT()) is specified, no other resource names can be specified.
X'00000010'	X'00004040'	The parameter override header has not been initialized. Retry the command after restart is complete.
X'00000014'	X'00005000'	Unable to get storage from IMODULE GETSTOR.
X'00000014'	X'00005010'	Unable to obtain latch.
X'00000014'	X'00005FFF'	Internal IMS error - should not occur.

The following table contains the completion code that can be returned on a UPDATE LE command.

Table 344. Completion code for the UPDATE LE command

Completion code	Meaning
0	The UPDATE LE command completed successfully for the specified resource.

Examples

The following are examples of the UPDATE LE command.

Example 1 for UPDATE LE command

Assume the following filters and parameters are specified on UPD LE SET(LERUNOPTS()) commands that are processed in the order listed.

1. TRAN(PART) LTERM(TERM2) SET(LERUNOPTS(hhhh))
2. TRAN(PART) LTERM(TERM2) SET(LERUNOPTS(iiii))
3. LTERM(TERM2) USERID(BETTY) SET(LERUNOPTS(gggg))
4. TRAN(PART) LTERM(TERM1) USERID(BOB) SET(LERUNOPTS(ffff))
5. TRAN(PART) LTERM(TERM1) USERID(BARBARA) SET(LERUNOPTS(eeee))
6. PGM(DFSSAM02) SET(LERUNOPTS(dddd))
7. TRAN(PART) LTERM(TERM1) SET(LERUNOPTS(cccc))
8. TRAN(PART) USERID(BETTY) SET(LERUNOPTS(bbbb))
9. TRAN(PART) PGM(DFSSAM02) SET(LERUNOPTS(aaaa))

Rules for matching an entry which results in an update of an existing entry:

- The number of filters defined on the UPDATE LE must match the number of filters defined in the entry.
- The filter values defined on the UPDATE LE must be an exact match for those defined in the entry.

The following table is a logical representation of the parameter override table entries at the end of the command processing. The table includes the transaction name, LTERM, USERID, Program, and LERUNOPTS for each entry.

Table 345. Parameter override table entries for UPDATE LE example 1

Entry#	TRAN	LTERM	USERID	PROGRAM	LERUNOPTS
1	PART			DFSSAM02	aaaa
2	PART		BETTY		bbbb
3	PART	TERM1			cccc
4				DFSSAM02	dddd
5	PART	TERM1	BARBARA		eeee
6	PART	TERM1	BOB		ffff
7		TERM2	BETTY		gggg
8	PART	TERM2			iiii

Example 2 for UPDATE LE command

TSO SPOC input:

```
UPD LE TRAN(IAPMDI26) USERID(USRT001)
SET(LERUNOPTS(RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR)))
```

TSO SPOC output:

```
MbrName    CC
SYS3        0
```

OM API input:

```
CMD(UPD LE TRAN(IAPMDI26) USERID(USRT001)
SET(LERUNOPTS(RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR))))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 17:56:10.220516</statime>
<stotime>2002.163 17:56:10.221547</stotime>
<staseq>B7C4CA4EDBFE420E</staseq>
<stoseq>B7C4CA4EDC3EB382</stoseq>
<rqsttkn1>USRT002 10105610</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>UPD </verb>
<kwd>LE              </kwd>
<input>UPD LE TRAN(IAPMDI26) USERID(USRT001)
 SET(LERUNOPTS(RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR)))
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"  llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT" align="right" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3    ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE LE command adds an entry to the LE runtime options table. The entry added by this command defines two filters: user ID and transaction. The transaction is set to IAPMDI26 and the user ID is set to USRT001. The runtime options string for this table entry is RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR). The output shows that IMS member SYS3 processed the command with a return code of 0.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

"QUERY LE command" on page 93

UPDATE MSLINK command

Use the UPDATE MSLINK command to set or change the MSC logical link attributes or to change the status of specified logical links.

The UPDATE MSLINK command can be either a type-1 or type-2 command.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 708
- "Usage notes" on page 713
- "Output fields" on page 714
- "Return, reason, and completion codes" on page 714
- "Examples" on page 716

Environment

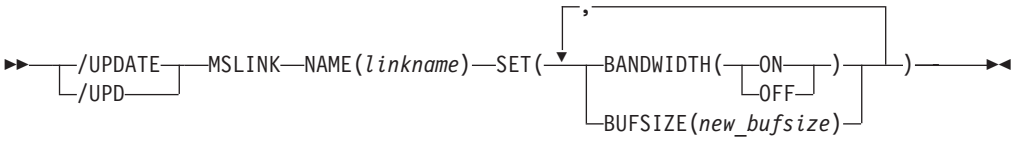
The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE MSLINK command and keywords can be issued.

Table 346. Valid environments for the UPDATE MSLINK command and keywords

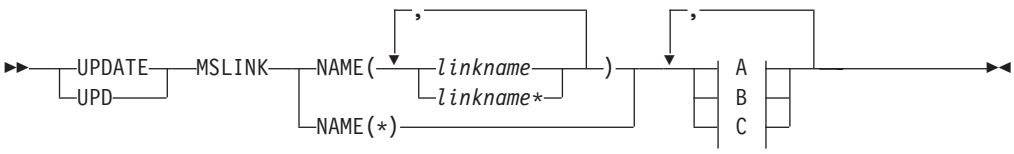
Command / keywords	DB/DC	DBCTL	DCCTL
UPDATE MSLINK	X		X
NAME	X		X
SET	X		X
START	X		X
STOP	X		X
OPTION	X		X

Syntax

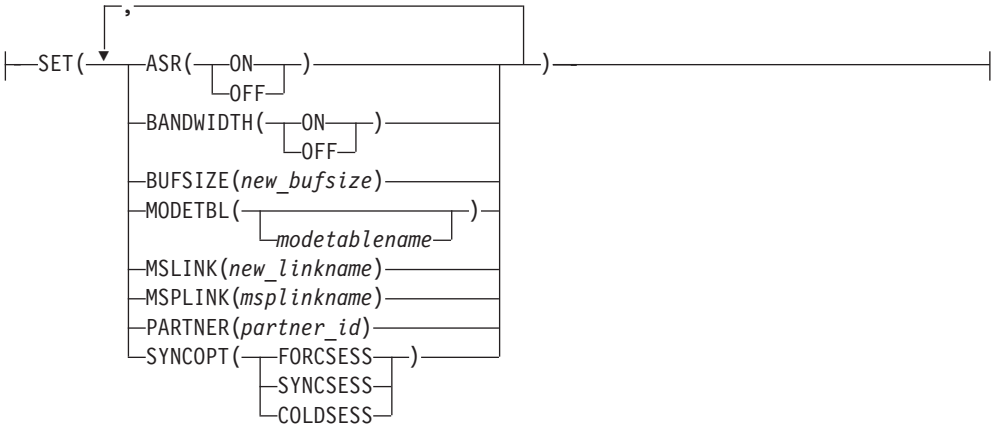
Type-1 command syntax



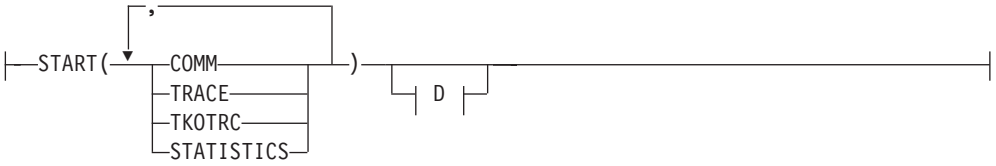
Type-2 command syntax



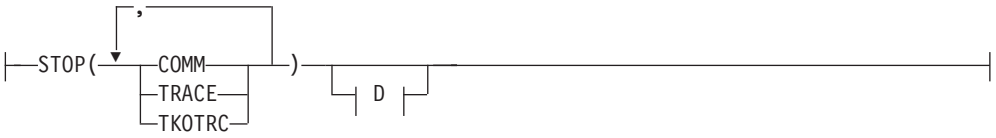
A:



B:



C:



D:

Keywords

The following keywords are valid for the UPDATE MSLINK command:

NAME()
Specifies the 1- to 8-character name of the MSC logical links to be processed. You can specify wildcard parameters (*) for the NAME keyword. The logical link names are processed if they match the wildcard name. NAME(*) and NAME(abc*) are examples of valid names.

SET()
Specifies the updates to attributes or parameters. You cannot specify both the SET() parameter and the STOP(COMM) parameter.

ASR()
The Automatic Session Restart (ASR) designation of a VTAM link changes when the Session Outage Notification facility of VTAM is used. The ASR parameter allows you to override the system definition option that is defined for the automatic session restart designation of a link.

You cannot change the designation of ASR for non-VTAM MSC links.

Stop the logical link before you change the ASR designation.

ON Specifies that the logical link uses automatic session restart. Automatic session restart is not necessarily activated for a link if a status of ASR is displayed for that link. You must also specify SONSCIP=YES on the APPL definition statement for VTAM when defining the network to VTAM. ASR works only if both sides of the link are using the ASR option.

OFF
Specifies that the logical link does not use automatic session restart.

BANDWIDTH()
Specifies whether the logical link uses bandwidth mode.

There is no system definition option to enable bandwidth mode. By default, logical links are initialized with bandwidth mode off.

To change the bandwidth mode, first stop the logical link. The link must not display a status ERE.

ON Specifies that the logical link uses bandwidth mode.

OFF
Specifies that the logical link does not use bandwidth mode.

BUFSIZE()
Changes the input and output buffer sizes for the logical link. The minimum buffer size is 1024, and the maximum buffer size is 65536.

The logical link must be stopped before changing the BUFSIZE value. Buffer sizes for logical links follow the same specification requirements as buffer sizes specified on the MSPLINK macro by using the BUFSIZE= keyword.

MODETBL()
The default mode table name of a link changes if you specify the MODETBL() parameter without the START(COMM) parameter. This default is usually established by system definition. To reset this field to its null state (as if a mode table name was not specified at system definition), specify MODETBL().

If you specify both the START(COMM) parameter with the MODETBL() parameter, then the specified mode table name is used only for the START(COMM) session. When the session is terminated, the link mode table name reverts to the default mode table name.

The mode table name determines the mode table entries to be used when activating a VTAM MSC session. You cannot use the MODETBL keyword with non-VTAM sessions.

The logical link must be stopped before changing the MODETBL value.

MSLINK()
Specifies the 1- to 8-alphanumeric new link name for the specified logical link. If you specify the MSLINK parameter, you can specify only one NAME parameter. The new link name cannot already exist as a logical link name. If the command is successful, the link name specified on the NAME keyword is no longer valid.

The logical link must be stopped before changing the MSLINK name.

MSPLINK()
Enter the 1- to 8-character name of the MSC physical link to which you want to assign this logical link. You assign the logical link to the physical link for input and output purposes. The physical link name specified must already exist.

The logical link must be stopped before changing the MSPLINK. If the logical link is being assigned to a VTAM physical link, then the physical link must also be stopped.

If the logical link is being assigned to a CTC or MTM link, then the logical link that is currently assigned to the target physical link must also be stopped.

PARTNER()
Specifies the 2-character alphabetic partner identification. This partner identification ensures that the two related logical links in two systems are always logically and physically connected. Both systems must have logical links with the same partner ID. The new partner ID specified must not already exist.

Only one logical link can be specified in the NAME keyword to change the PARTNER keyword.

The logical link must be stopped before changing the PARTNER keyword.

SYNCOPT()
Specifies the system definition option to be overridden for the named logical link. This keyword allows the user to override the system definition option defined for forcing resynchronization until the next UPDATE MSLINK command is issued or IMS is cold started.

SYNCOPT() is valid only for VTAM links.

The logical link must be stopped before changing SYNCOPT.

FORCSESS
Specifies that the session initiation is to be completed regardless of the agreement between session restart modes and the message sequence numbers. Use of the FORCSESS parameter could cause messages to be lost or duplicated.

SYNCSESS
Specifies that the session initiation is to be completed only when the session restart modes and message sequence numbers agree.

COLDSESS
Resets the session restart mode to COLD by resetting all the MSLINK dynamic flags and fields to zero, which allows cold session initiation of the link. Use the COLDSESS parameter when attempts to warm start the session fail.

Use of the COLDSESS parameter could cause messages to be lost or duplicated.

Before issuing the command with the COLDSESS option, stop the link first. When COLDSESS is specified, it should be used on both sides of the link.

The COLDSESS parameter does not override the current setting of SYNCSESS or FORCSESS.

START()
Starts the specified MSC logical links, or starts tracing of the specified logical links.

COMM
Starts the specified MSC logical links and allows IMS to start sending and receiving messages.

Communication between IMS systems does not begin until an UPDATE MSLINK command is entered in both systems for CTC or MTM links, or in one of the systems for a VTAM link.

The COMM parameter cannot be specified in both START() and STOP(). The logical link must be stopped and idle, and the assigned physical link must be open, as shown in the QUERY command output.

TRACE
Turns on the tracing of internal IMS events that are related to the specified logical links. Unlike the equivalent type-1 /TRACE command, the trace level cannot be specified on the UPDATE command. Instead, IMS uses the default level of 4. The status and options of the current IMS traces can be displayed with the QUERY MSLINK command. TRACE cannot be specified in both START() and STOP().

Note: The type-2 command UPDATE MSLINK NAME(*linkname*) START(TRACE) uses the same level and module settings that were used the last time the /TRACE SET (ON) LINK command was issued. If a /TRACE SET (ON) LINK command has not been issued since the last cold start, this command defaults to MODULE=ALL and LEVEL=4.

TKOTRC
Enables tracing during takeover. This tracing is separate from regular tracing. This keyword applies only in an XRF environment. TRACE and TKOTRC cannot both be specified under START(). TKOTRC cannot be specified in both START() and STOP().

STATISTICS
Provides statistics support to MSC logical links. These statistics can be used to determine the link performance, adjust the link and message queue buffer sizes to optimum sizes, and increase the link performance. Each logical link (MSLINK) has a work area where statistics are kept.

STOP()
Stops the specified MSC logical links or stops tracing of the specified logical links.

COMM
Stops the specified MSC logical links, and allows IMS to stop sending and receiving messages. The partner link in another IMS system stops itself. COMM cannot be specified in both START() and STOP(). You cannot specify both STOP(COMM) and SET() together.

TRACE
Turns off the tracing of internal IMS events that are related to the specified

logical links. The status and options of the current IMS traces can be displayed with the QUERY MSLINK command. TRACE cannot be specified in both START() and STOP().

TKOTRC()
Disables tracing during takeover. This tracing is separate from regular tracing. This keyword applies only in an XRF environment. TKOTRC cannot be specified in both START() and STOP().

OPTION()
Specifies options for the UPDATE MSLINK command. OPTION is valid only with START(STATISTICS) or STOP(COMM).

FORCE
Can be specified only with STOP(COMM) for VTAM links. Use the FORCE parameter when an MSC link will not clean up and idle during normal UPDATE STOP(COMM) processing, even though VTAM terminated the session. You can use the FORCE parameter with some network commands, to clean up and idle the VTAM link within IMS, so that you can issue the UPDATE MSLINK command with the NAME(linkname | *) START(COMM) options to restart the link.

RESET | NORESET
Issuing the UPDATE MSLINK() START(STATISTICS) OPTION(RESET) command will reset the link statistics and set the start time to the current time. The logical link does not need to be stopped and idle to reset the statistics values, or the reset mode, as is the case for changing many of the link characteristics, such as buffer size and bandwidth mode.

OPTION(RESET), OPTION(CHKPT,RESET), and OPTION(CHKPT, NORESET) can be specified with START(STATISTICS).

RESET
When specified without CHKPT, this indicates that statistics for the specified logical link should be reset immediately. When specified with CHKPT, this indicates that statistics for the specified logical link should be reset at every system checkpoint. This only applies to statistics reported on the QUERY MSLINK command. Statistics reported in the log records during system checkpoints are cumulative over the life of IMS, and are not reset by the UPDATE MSLINK command.

NORESET
When specified with CHKPT, this indicates that statistics for the specified logical link should not be reset at any system checkpoint. This only applies to statistics reported on the QUERY MSLINK command. Statistics reported in the log records during system checkpoints are cumulative over the life of IMS, and are not reset by the UPDATE MSLINK command.

CHKPT
When specified with RESET, this indicates that statistics for the specified logical link should be reset at every system checkpoint. When specified with NORESET, this indicates that statistics for the specified logical link should not be reset at any system checkpoint. This only applies to statistics reported on the QUERY MSLINK command. Statistics reported in the log records during system checkpoints are cumulative over the life of IMS, and are not reset by the UPDATE MSLINK command.

OPTION(CHKPT,RESET) and OPTION(CHKPT, NORESET) can be specified with START(STATISTICS).

Usage notes

You can issue the type-2 UPDATE MSLINK command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for the UPDATE MSLINK command is defined in XML and the syntax is available to automation programs that communicate with OM.

The UPDATE MSLINK command compared to other commands

The following table shows variations of the UPDATE MSLINK command and the type-1 IMS commands that perform similar functions.

Table 347. Type-1 equivalents for the UPDATE MSLINK command

UPDATE MSLINK command	Similar IMS command
UPDATE MSLINK NAME(linkname | linkname* | *) SET(ASR(ON|OFF))	/CHANGE LINK link # | ALL ASR ON | OFF
UPDATE MSLINK NAME(linkname | linkname* | *) SET(MODETBL(modetablename))	/CHANGE LINK link # | ALL MODE(modename | NONE)
UPDATE MSLINK NAME(linkname) SET(MSLINK(linkname))	No similar type-1 IMS command exists.
UPDATE MSLINK NAME(linkname | linkname* | *) SET(MSPLINK(msplinkname))	/MSASSIGN LINK link # MSPLINK msplinkname
UPDATE MSLINK NAME(linkname | linkname* | *) SET(PARTNER(partner_id))	No similar type-1 IMS command exists.
UPDATE MSLINK NAME(linkname | linkname* | *) SET(SYNCOPT(FORCSESS))	/CHANGE LINK link#|ALL FORCSESS
UPDATE MSLINK NAME(linkname | linkname* | *) SET(SYNCOPT(SYNCSESS))	/CHANGE LINK link# | ALL SYNCSESS
UPDATE MSLINK NAME(linkname | linkname* | *) SET(SYNCOPT(COLDSESS))	/CHANGE LINK link# | ALL COLDSESS
UPDATE MSLINK NAME(linkname | linkname* | *) START(COMM)	/RSTART LINK link # | ALL
UPDATE MSLINK NAME(linkname | linkname* | *) SET(MODETBL(modetablename)) START(COMM)	/RSTART LINK link# | ALL MODE modename
UPDATE MSLINK NAME(linkname | linkname* | *) START(TRACE)	/TRACE SET ON | OFF LINK link # | ALL
UPDATE MSLINK NAME(linkname | linkname* | *) START(TKOTRC)	/TRACE SET ON LINK link # | ALL TAKEOVER
UPDATE MSLINK NAME(linkname | linkname* | *) STOP(COMM)	/PSTOP LINK link # | ALL
UPDATE MSLINK NAME(linkname | linkname* | *) STOP(COMM) OPTION(FORCE)	/PSTOP LINK link # PURGE or /PSTOP LINK link # FORCE
UPDATE MSLINK NAME(linkname | linkname* | *) STOP(COMM,TRACE) OPTION(FORCE)	/PSTOP LINK link # FORCE /TRA SET OFF LINK link#

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 348. Output field descriptions for the UPDATE MSLINK command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
LINKN	N/A	Logical link number.
MBR	N/A	IMSplex member that built the output line.
MSL	N/A	Logical link name.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE MSLINK command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 349. Return and reason codes for the UPDATE MSLINK command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE MSLINK command completed successfully.
X'00000008'	X'0000200C'	No resources to be updated were found. The resource names specified might be invalid or no resources match the specified filter.
X'00000008'	X'00002040'	More than one filter value is specified on the UPDATE MSLINK command.
X'00000008'	X'00002044'	The UPDATE MSLINK command is not processed because the same attribute value was specified for the START and STOP parameters.
X'00000008'	X'00002048'	Invalid SET attribute is specified.
X'00000008'	X'00002133'	The UPDATE MSLINK command is not processed because multiple logical link names were specified, which is invalid for the SET filters specified in the command.
X'00000008'	X'00002134'	TheUPDATE MSLINK command specified SET(MSPLINK(*msplinkname*)), but *msplinkname* does not exist.

Table 349. Return and reason codes for the UPDATE MSLINK command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'00004000'	Command is not valid during IMS restart.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000402C'	Command is not valid on the non-MSC-capable system.
X'00000014'	X'00005004'	The UPDATE MSLINK command processing terminated as a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005010'	A latch could not be obtained.

Errors that are unique to the processing of the UPDATE MSLINK command are returned as completion codes. A completion code is returned for each action against an individual resource.

If the UPDATE MSLINK command is entered as a type-2 command, a message that contains the completion codes will be issued. The codes listed in the following table are for a type-1 UDPATE MSLINK command.

Table 350. Completion codes for the UPDATE MSLINK command

Completion code	Completion code text	Meaning
0		The UPDATE MSLINK command completed successfully for the resource.
10	NO RESOURCES FOUND	MSLINK name is invalid, or the specified wildcard parameter does not match any resource names.
11	RESOURCE ALREADY EXISTS	The linkname specified in SET(MSLINK(linkname)) already exists as a logical link.
62	HIOP STORAGE ERROR	IMS was unable to obtain storage from the HIOP pool.
8D	RESOURCE IS NOT STOPPED	The logical link is not stopped. The link must be stopped for the updates specified.
100	INV SET KEYWORD FOR LINK TYPE	A keyword specified in SET() is invalid for the corresponding physical link type.
101	TARGET MSPLINK NOT STOPPED	The target physical link specified in SET(MSPLINK()) is not stopped.

Table 350. Completion codes for the UPDATE MSLINK command (continued)

Completion code	Completion code text	Meaning
102	TARGET MSLINK NOT STOPPED	The logical link that is currently assigned to the target physical link specified in SET(MSPLINK()) is not stopped.
103	NEW PARTNER ID ALREADY EXISTS	The new partner ID specified in SET(PARTNER()) already exists.
104	MSLINK NOT ASSIGNED TO MSPLINK	The update cannot be performed because the logical link is not assigned to a physical link.
105	MSPLINK NOT OPEN	The update cannot be performed because the assigned physical link is not open.
106	INVALID OPTION FOR LINK TYPE	A keyword specified in OPTION() is invalid for the corresponding physical link type.
107	OPTION(FORCE) ALREADY IN EFFECT	OPTION(FORCE) has already been invoked for the logical link.

Examples

The following are examples of the UPDATE MSLINK command:

Example 1 for UPDATE MSLINK command

TSO SPOC input:

```
UPD MSLINK NAME(STAR1L, STAR2L, STAR3L*) SET(SYNCOPT(FORCSESS),
ASR(OFF),MODETBL(),MSPLINK(STAR1)) START(COMM,TRACE)
```

TSO SPOC output:

```
MSLink    MbrName LinkNum CC CCText
STAR1L    IMSA    12      0
STAR2L    IMSA            10 NOT FOUND
STAR3L    IMSA    5       0
STAR3L1   IMSA    6       0
STAR3L2   IMSA    7       0
```

Explanation: This UPDATE MSLINK command is issued to do the following:

1. Override the system definition option for the named logical links: STAR1L and logical links matching STAR3L* (STAR3L, STAR3L1, and STAR3L2). STAR2L is unknown.
2. Change the automatic session restart designation to OFF for the specified logical links.
3. Reset the mode table name of the specified logical links to NONE.
4. Assign the specified logical links to the physical link: STAR1.
5. Allow IMS to start sending and receiving messages on the specified links.
6. Enable tracing of control block trace information.

Example 2 for UPDATE MSLINK command

TSO SPOC input:

```
UPDATE MSLINK NAME(LNK12V02) SET(BANDWIDTH(ON), BUFSIZE(4096))
```

TSO SPOC output:

```
MSLink            MSLink#    MbrName    CC
LNK12V02               10    IMS1        0
```

Explanation: This command changes the buffer size to 4096 and turns on the bandwidth.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related tasks:

Diagnosing link problems by using MSC link statistics (Diagnosis)

Related reference:

Command keywords and their synonyms (Commands)

List of commands with similar functions for multiple resources (Operations and Automation)

UPDATE MSNAME command

Use the UPDATE MSNAME command to set or change the MSC logical link path attributes or to change the status of specified logical link paths.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 718
- "Usage notes" on page 720
- "Equivalent IMS type-1 commands" on page 720
- "Output fields" on page 720
- "Return, reason, and completion codes" on page 721
- "Examples" on page 723

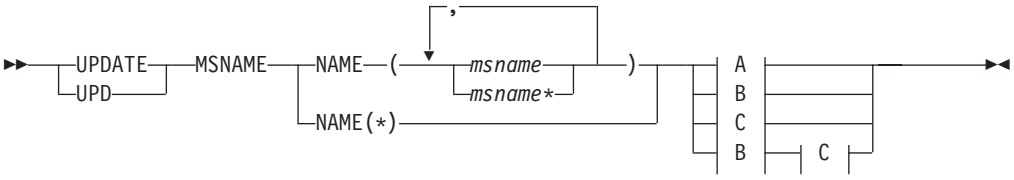
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE MSNAME command and keywords can be issued.

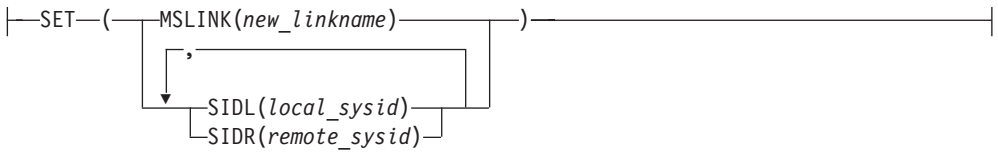
Table 351. Valid environments for the UPDATE MSNAME command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
UPDATE MSNAME	X		X
NAME	X		X
SET	X		X
START	X		X
STOP	X		X

Syntax



A:



B:

C:

Keywords

The following keywords are valid for the UPDATE MSNAME command:

NAME()
Specifies the 1- to 8-character name of the MSC logical link path to be processed. You can specify wildcards (*) for the NAME keyword. NAME(*) and NAME(abc*) are examples of valid names.

SET()
Specifies the updates to attributes or parameters. You can only change attributes for statically-defined MSNAMEs. Dynamic MSNAMEs cannot be changed.

MSLINK()
Assigns the specified logical link path to the logical link for input and output purposes. Enter the 1- to 8-character name of the MSC logical link to which you want to assign this logical link path. The logical link name specified must already exist.

Both the source logical link (where the MSNAME is currently assigned) and the target logical link (where the MSNAME is being assigned to) must be stopped.

SIDL()
Specifies the system identification of a local system in a multiple system configuration. The specified value must be a number from 1 to 2036. The

new specified value must be currently undefined or already defined as a local SID on this system. The SID cannot already be defined as remote.

You can change the SIDL value for only one logical link path on a command. Therefore, if you specify the SIDL parameter, you can specify only one NAME parameter.

The logical link and logical link path must be stopped to change the SIDL.

To change the SIDR or SIDL value of a logical link path:

1. Verify the existing SID values of the logical link path of either /DIS ASMT MSNAME or QUERY MSNAME SHOW(SYSID).

   For SIDL verification only, the operator must verify that the current SIDL value is not the lowest SIDL value in the system. If it is the lowest SIDL value in the system, and there are no other logical link paths in the system with the same SIDL, then the SIDL cannot be changed because the lowest SIDL in the system cannot change.
2. Verify there are no transactions with the existing SID values of either /DIS SYSID TRAN or QUERY TRAN SHOW(MSNAME).
3. If any transactions were found in step 2, assign the transactions to a new valid system of either /MSASSIGN TRAN TO MSNAME, UPDATE TRAN SET(MSNAME()) or UPDATE TRAN SET(SIDR(),SIDL()).
4. Change the SID values of the logical link path using UPDATE MSNAME SET(SIDR(),SIDL()). However, this is not allowed if the new SIDL value is lower than the current lowest SIDL in the system. The lowest SIDL in the system cannot change, and must remain the lowest SIDL in the system.

SIDR()

Specifies the system identification of a remote system in a multiple system configuration. The value specified must be a number from 1 to 2036. The new value specified must be currently undefined on this system (it cannot be already defined as remote or local).

You can change the SIDR value for only one logical link path on a command. Therefore, if you specify the SIDR parameter, you can specify only one NAME parameter.

The logical link and logical link path must be stopped to change the SIDR.

To change the SIDR or SIDL value of a logical link path:

1. Verify the existing SID values of the logical link path of either /DIS ASMT MSNAME or QUERY MSNAME SHOW(SYSID).
2. Verify there are no transactions with the existing SID values of either /DIS SYSID TRAN or QUERY TRAN SHOW(MSNAME).
3. If any transactions were found in step 2, assign the transactions to a new valid system of either /MSASSIGN TRAN TO MSNAME, UPDATE TRAN SET(MSNAME()) or UPDATE TRAN SET(SIDR(),SIDL()).
4. Change the SID values of the logical link path using UPDATE MSNAME SET(SIDR(),SIDL()).

START()

Specifies the attributes to start. You cannot specify the START and STOP parameters with the same attributes.

Q Starts the queuing to the logical link path and starts input for the specified logical link path.

SEND
Starts the sending of messages to the specified logical link path.

STOP()
Specifies the attributes to stop. You cannot specify the START and STOP parameters with the same attributes.

Q Stops the queuing to the logical link path and stops input for the specified logical link path.

SEND
Stops the sending of messages to the specified logical link path.

Usage notes

You can issue this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE MSNAME command and the type-1 IMS commands that perform similar functions.

Table 352. Type-1 equivalents for the UPDATE MSNAME command

UPDATE MSNAME command	Similar IMS type-1 command
UPDATE MSNAME NAME(msname) SET(MSLINK(linkname))	/MSASSIGN MSNAME msname TO LINK link #
UPDATE MSNAME NAME(msname) SET(SIDR(remote_sysid))	No similar type-1 IMS command exists.
UPDATE MSNAME NAME(msname) SET(SIDL(local_sysid))	No similar type-1 IMS command exists.
UPDATE MSNAME NAME(msname | msname* | *) START(Q,SEND)	/START MSNAME msname | msname* | ALL
UPDATE MSNAME NAME(msname | msname* | *) STOP(Q,SEND)	/STOP MSNAME msname | msname* | ALL
UPDATE MSNAME NAME(msname | msname* | *) STOP(Q),START(SEND)	/PURGE MSNAME msname | msname* | ALL

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 353. Output field descriptions for the UPDATE MSNAME command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
MBR	N/A	IMSplex member that built the output line.
MSN	N/A	Logical link path name.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE MSNAME command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 354. Return and reason codes for the UPDATE MSNAME command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE MSNAME command completed successfully.
X'00000008'	X'0000200C'	No resources to be updated were found. The resource names specified might be invalid or no resources match the specified filter.
X'00000008'	X'00002040'	More than one filter value is specified on the UPDATE MSNAME command.
X'00000008'	X'00002044'	The UPDATE MSNAME command is not processed because the same attribute value was specified for the START and STOP parameters.
X'00000008'	X'00002126'	The value of SIDL() is invalid. Valid values are 1-2036.
X'00000008'	X'00002128'	The value of SIDR() is invalid. Valid values are 1-2036.
X'00000008'	X'00002136'	The name specified in MSLINK() is not a valid logical link name.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'00004000'	Command is not valid during IMS restart.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.

Table 354. Return and reason codes for the UPDATE MSNAME command (continued)

Return code	Reason code	Meaning
X'00000010'	X'0000402C'	Command is not valid on the non-MSC-capable system.
X'00000014'	X'00005004'	The UPDATE MSNAME command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005010'	A latch could not be obtained.

Errors that are unique to the processing of the UPDATE MSNAME command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 355. Completion codes for the UPDATE MSNAME command

Completion code	Completion code text	Meaning
0		The UPDATE MSNAME command completed successfully for the resource.
10	NO RESOURCES FOUND	MSNAME name is invalid, or the specified wildcard parameter does not match any resource names.
45	INVALID SIDR VALUE	The value specified in SIDR() is invalid. The value must be currently undefined in this system.
83	INVALID SIDL VALUE	The value specified in SIDL() is invalid. The value must be currently undefined or local in this system.
B7	USE MANAGER ERROR	IMS encountered an internal error using the USE manager.
102	TARGET MSLINK NOT STOPPED	The update cannot complete because the link specified in MSLINK() is not stopped.
109	INVALID FOR DYNAMIC MSNAME	The update cannot complete because the logical link path specified in NAME() is dynamic. The update is not valid for dynamic logical link paths that are built in a shared queues system, which represent logical link paths defined only on other IMS systems in the shared queues group.
10A	SOURCE MSLINK NOT STOPPED	The update cannot complete because the logical link to which the specified logical link path is assigned is not stopped.
10B	MSC RESOURCE IS IN USE	The update cannot complete because there is in-doubt message activity for a remote transaction or lterm related to the logical link path.
10D	INVALID SIDL, LESS THAN LOW SIDL	The value specified in SIDL() is invalid. The value cannot be lower than the current lowest local SID in this system.

Table 355. Completion codes for the UPDATE MSNAME command (continued)

Completion code	Completion code text	Meaning
10E	LAST MSNAME FOR LOW SIDL	The current local SIDL value cannot be changed for this logical link path because the current SIDL value is the lowest local SIDL value, and this is the only logical link path that has this local SIDL value.
10F	SIDL STILL LOCAL IN REMOTE TRAN	The current local SIDL value cannot be changed for this logical link path because there are remote transactions defined with this local SIDL value, and this the only logical link path that has this local SIDL value.
110	SIDR STILL LOCAL IN REMOTE TRAN	The current remote SIDR value cannot be changed for this logical link path because there are remote transactions defined with this remote SIDR value.

Examples

The following are examples of the UPDATE MSNAME command:

Example 1 for UPDATE MSNAME command

TSO SPOC input: `UPD MSNAME NAME(STAR1N1,STAR2N1) STOP(Q,SEND)`

TSO SPOC output:

```
MSName  MbrName CC
STAR1N1 IMSA    0
STAR2N1 IMSA    0
```

Explanation: This UPDATE MSNAME command is issued to stop queuing and the sending of messages to the named logical link paths. This allows the attributes to be changed by another UPD MSNAME command.

Example 2 for UPDATE MSNAME command

TSO SPOC input:

```
UPD MSNAME NAME(STAR1N1) SET(SIDR(10),SIDL(20))
```

TSO SPOC output:

```
MSName  MbrName CC
STAR1N1 IMSA    0
```

Explanation: This UPDATE MSNAME command is issued to do the following:

- Identify the remote system that is represented by SIDR(10).
- Identify the local system as SIDL(20) for routing messages back to this system.

Example 3 for UPDATE MSNAME command

TSO SPOC input:

```
UPD MSNAME NAME(STAR1N1) SET(MSLINK(STAR3L))
```

TSO SPOC output:

```
MSName  MbrName CC
STAR1N1 IMSA    0
```

Explanation: This UPDATE MSNAME command is issued to assign the specified logical link path STAR1N1 to STAR3L.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

/MSASSIGN command (Commands)

List of commands with similar functions for multiple resources (Operations and Automation)

"/START MSNAME command" on page 481

"/STOP MSNAME command" on page 527

Chapter 4, "/PURGE command," on page 29

UPDATE MSPLINK command

Use the UPDATE MSPLINK command to set or change the MSC physical link attributes or to change the status of the specified physical links.

Subsections:

- "Environment"
- "Syntax" on page 725
- "Keywords" on page 725
- "Usage notes" on page 726
- "Equivalent IMS type-1 commands" on page 726
- "Output fields" on page 727
- "Return, reason, and completion codes" on page 727
- "Examples" on page 728

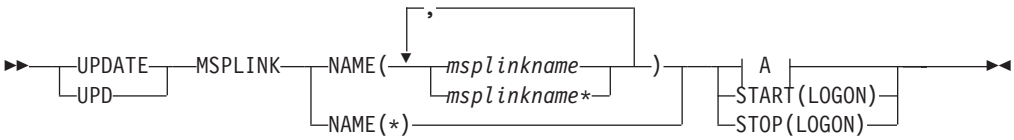
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE MSPLINK command and keywords can be issued.

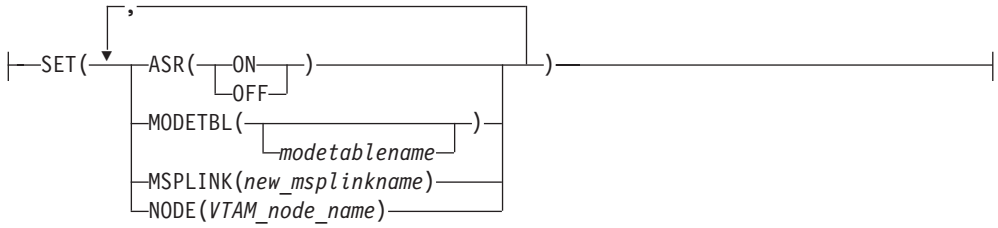
Table 356. Valid environments for the UPDATE MSPLINK command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
UPDATE MSPLINK	X		X
NAME	X		X
SET	X		X
START	X		X
STOP	X		X

Syntax



A:



Keywords

The following keywords are valid for the UPDATE MSPLINK command:

NAME()
Specifies the 1- to 8-character name of the MSC physical link that is processed. You can specify wildcards (*) in the name. NAME(*) and NAME(abc*) are examples of valid names.

SET()
Specifies the attribute values to be updated or sets the resource state.

ASR(), MODETBL() and NODE() are VTAM link specific parameters.

ASR()
The Automatic Session Restart (ASR) designation for each logical link assigned to the specified VTAM physical link changes when the Session Outage Notification facility of VTAM is used. The ASR parameter allows you to override the system definition option that is defined for the automatic session restart designation of a link.

You cannot change the designation of ASR for non-VTAM MSC links.

Stop the physical link and each assigned logical link before you change the ASR designation.

ON Specifies that the logical link uses automatic session restart. Automatic session restart is not necessarily activated for a link if a status of ASR is displayed for that link. SONSCIP=YES must also be specified on the APPL definition statement for VTAM when defining the network to VTAM. ASR works only if both sides of the link use the ASR option.

OFF
Specifies that the logical link does not use automatic session restart.

MODETBL()
Changes the name of the default VTAM logon mode table entry (logon mode table) for this physical link. This changes the name for each logical link currently assigned to this physical link. This name is usually

established by system definition. To reset this field to its null state (as if no mode table name was specified at system definition), specify MODETBL().

You cannot use the SET(MODETBL) parameter for a non-VTAM MSC link.

The physical link and each assigned logical link must be stopped before changing the MODETBL.

MSPLINK()
Specifies the 1- to 8-alphanumeric new physical link name for this physical link. This keyword also allows you to change the name of a reserved link to a real name and activate it. The name specified in MSPLINK() must be unique. If the command is successful, the old physical link name specified on the NAME keyword is no longer valid.

Only one NAME parameter can be specified to change the MSPLINK name.

The physical link must be stopped before changing the MSPLINK.

NODE()
Specifies a 1- to 8-alphanumeric name for the remote VTAM node (APPLID) name for the VTAM line. This node is the VTAM node name of the remote IMS system at the other end of the link. This node is the label on the VTAM APPL statement for the remote IMS system. For an MSC VTAM link communicating with an XRF complex, the node name must be the VTAM USERVAR that is associated with the partner XRF complex.

The physical link and each assigned logical link must be stopped before changing the NODE.

START()
Specifies the attributes to start.

LOGON
Only MSC VTAM link are reset to enable logons. START(LOGON) and STOP(LOGON) are mutually exclusive. START(LOGON) and STOP(LOGON) are mutually exclusive.

STOP()
Specifies the attributes to stop.

LOGON
Stops logons to the physical link (only for MSC VTAM links).

The STOP(LOGON) does not affect links that are already in session.

Usage notes

You can issue this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE MSPLINK command and the type-1 IMS commands that perform similar functions.

Table 357. Type-1 equivalents for the UPDATE MSPLINK command

UPDATE MSPLINK command	Similar IMS type-1 command
UPDATE MSPLINK NAME(*msplinkname*) SET(ASR(ON | OFF))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname*) SET(MODETBL(*modetablename*))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname*) SET(MSPLINK(*new_msplinkname*))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname*) SET(NODE(*VTAM_node_name*))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname* | *) START(LOGON)	/RSTART MSPLINK *msplinkname* | ALL
UPDATE MSPLINK NAME(*msplinkname* | *) STOP(LOGON)	/PSTOP MSPLINK *mplinkname* | ALL

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 358. Output field descriptions for the UPDATE MSPLINK command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
MBR	N/A	IMSplex member that built the output line.
MSPL	N/A	Physical link name.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE MSPLINK command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 359. Return and reason codes for the UPDATE MSPLINK command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE MSPLINK command completed successfully.
X'00000008'	X'00002040'	More than 1 filter value is specified on the UPDATE MSPLINK command.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000014'	X'00005004'	The UPDATE MSPLINK command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.

Errors that are unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 360. Completion codes for the UPDATE MSPLINK command

Completion code	Completion code text	Meaning
0		The UPDATE MSPLINK command completed successfully for the resource.
10	NO RESOURCES FOUND	MSPLINK name is invalid, or the specified wildcard parameter does not match any resource names.
11		The link name specified in MSPLINK() already exists as a physical link.
8D		The physical link is not stopped. The link must be stopped for the updates specified.

Examples

The following are examples of the UPDATE MSPLINK command:

Example 1 for UPDATE MSPLINK command

TSO SPOC input:

```
UPD MSPLINK NAME(STAR1) SET(MODETBL(LOGON12B), NODE(APPL12B), ASR(ON))
```

TSO SPOC output:

```
MSName   MbrName CC
STAR1    IMSA     0
```

Explanation: This UPDATE MSPLINK command is issued to do the following:

1. Change the VTAM logon mode table entry to LOGON12B.
2. Change the VTAM node name of the remote system at the other end of the link to APPL12B.
3. Override the automatic session restart definitions established by system definition.

Example 2 for UPDATE MSPLINK command

TSO SPOC input:

```
UPD MSPLINK NAME(STAR1, STAR2, STAR2B*) START(LOGON)
```

TSO SPOC output:

```
MSName   MbrName CC   CCTEXT
STAR1    IMSA    0
STAR2    IMSA    10   NOT FOUND
STAR2B   IMSA    0
STAR2B11 IMSA    0
STAR2B12 IMSA    0
```

Explanation: This command enables logon for the specified physical links: STAR1, STAR2B, STAR2B11, and STAR2B12. STAR2 is unknown.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

List of commands with similar functions for multiple resources (Operations and Automation)

UPDATE ODBM commands

Use the UPDATE ODBM commands to change Open Database Manager (ODBM) settings.

Subsections:

- "UPDATE ODBM START(CONNECTION) command"
- "UPDATE ODBM START(TRACE) command" on page 733
- "UPDATE ODBM STOP(CONNECTION) command" on page 735
- "UPDATE ODBM STOP(TRACE) command" on page 738
- "UPDATE ODBM TYPE(CONFIG) command" on page 741

UPDATE ODBM START(CONNECTION) command

Use the UPDATE ODBM START(CONNECTION) command to start connections to data stores and aliases for Open Database Manager (ODBM).

Subsections:

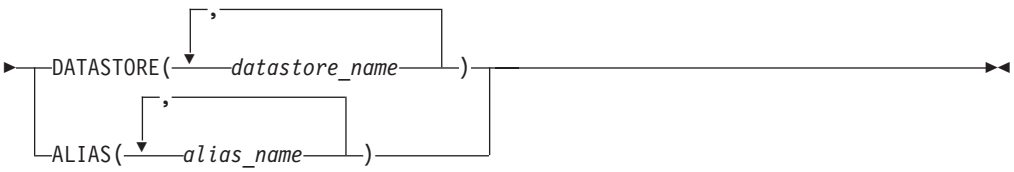
- "Environment" on page 730

- "Syntax"
- "Keywords"
- "Usage notes"
- "Output fields" on page 731
- "Return, reason, and completion codes" on page 731
- "Examples" on page 732

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the UPDATE ODBM START(CONNECTION) command.

ALIAS()
Specifies the 1–4 character alias name. Wildcards can be specified for the alias name. You can specify more than one alias name.

There is no default for ALIAS(). Specifying the ALIAS keyword will start ODBM connections to all the aliases specified with *alias_name*. ALIAS() and DATASTORE() are mutually exclusive.

DATASTORE()
Specifies the 1–4 character data store name. Wildcards can be specified for the data store name. You can specify more than one data store names.

There is no default for DATASTORE(). Specifying the DATASTORE keyword will start ODBM connections to all the data stores specified with *datastore_name* as well as their associated aliases. DATASTORE() and ALIAS() are mutually exclusive.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 361. Output field descriptions for the UPDATE ODBM START(CONNECTION) command

Short label	Keyword	Meaning
ALIAS	ALIAS	The 1–4 character alias name that is associated with an IMS data store connection.
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DSTR	N/A	Data store name.
MBR	N/A	Name of the ODBM member that processed the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE ODBM START(CONNECTION) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 362. Return and reason codes for the UPDATE ODBM START(CONNECTION) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPD ODBM START(CONNECTION) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 363. Completion codes for the UPDATE ODBM START(CONNECTION) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM START(CONNECTION) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
3	DATASTORE NOT AVAILABLE	An attempt to start a data store connection failed. The data store is unavailable.
4	DATASTORE CONNECTION FAILED	An attempt to start a data store connection failed. Message CSL4006W accompanies this completion code and explains the AIB return code, AIB reason code, and AIB error extension. (See note)
5	DATASTORE NOT RRS CAPABLE	An attempt to start a data store connection failed. The data store is not RRS capable.
10	NO RESOURCES FOUND	No resources were found.

Note: The command response is encapsulated in <rsp> and </rsp> tags. Message CSL4006W is returned within <msg> and </msg> tags.

Examples

Example 1 for UPDATE ODBM START(CONNECTION) command

TSO SPOC input:

```
UPD ODBM START(CONNECTION) DATASTORE(*)
```

TSO SPOC output:

```
MbrName     DatastoreName   CC
ODBM01OD    IMS1             0
ODBM01OD    IMS2             0
ODBM02OD    IMS3             0
```

Explanation: The UPDATE command will start connections with all data stores that are known to ODBM.

Example 2 for UPDATE ODBM START(CONNECTION) command

TSO SPOC input:

```
UPD ODBM START(CONNECTION) ALIAS(IO1A,IO1B)
```

TSO SPOC output:

```
MbrName     AliasName    CC      DatastoreName
ODBM01OD    IO1A          0      IMS1
ODBM01OD    IO1B          0      IMS1
```

Explanation: The UPDATE command will start ODBM connections with the ODBM members whose alias names are either IO1A or IO1B.

UPDATE ODBM START(TRACE) command

Use the UPDATE ODBM START(TRACE) command to request that ODBM traces be started.

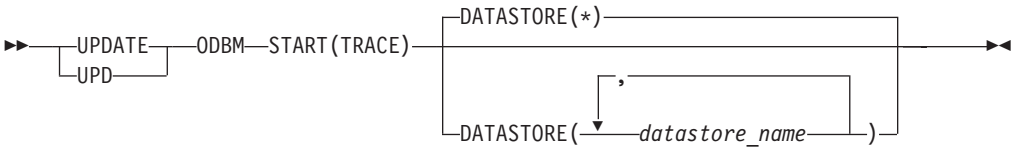
Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Output fields"
- "Return, reason, and completion codes" on page 734
- "Examples" on page 735

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the UPDATE ODBM START(TRACE) command.

DATASTORE()
Specifies the 1–4 character data store name. Wildcards can be specified for the data store name. The *datastore_name* is a repeatable parameter. The default is DATASTORE(*), which applies to all data store names that are known to ODBM.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 364. Output field descriptions for the UPDATE ODBM START(TRACE) command

Short label	Keyword	Meaning
ALIAS	N/A	Alias name.
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DSTR	N/A	Data store name.
MBR	N/A	Name of the ODBM member that processed the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE ODBM START(TRACE) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 365. Return and reason codes for the UPDATE ODBM START(TRACE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE ODBM START(TRACE) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 366. Completion codes for the UPDATE ODBM START(TRACE) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM START(TRACE) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for UPDATE ODBM START(TRACE) command

TSO SPOC input:

```
UPD ODBM START(TRACE) DATASTORE(IMS1,IMS2)
```

TSO SPOC output:

```
MbrName    DatastoreName    CC
ODBM010D   IMS1              0
ODBM010D   IMS2              0
```

Explanation: The UPDATE command will start ODBM traces for data stores IMS1 and IMS2.

Example 2 for UPDATE ODBM START(TRACE) command

TSO SPOC input:

```
UPD ODBM START(TRACE) DATASTORE(*)
```

TSO SPOC output:

```
MbrName    DatastoreName   CC
ODBM010D    IMS1            0
ODBM010D    IMS2            0
ODBM020D    IMS3            0
```

Explanation: The UPDATE command will start ODBM traces for all data stores that are known to ODBM.

UPDATE ODBM STOP(CONNECTION) command

Use the UPDATE ODBM STOP(CONNECTION) command to stop connections to data stores and aliases for Open Database Manager (ODBM).

Subsections:

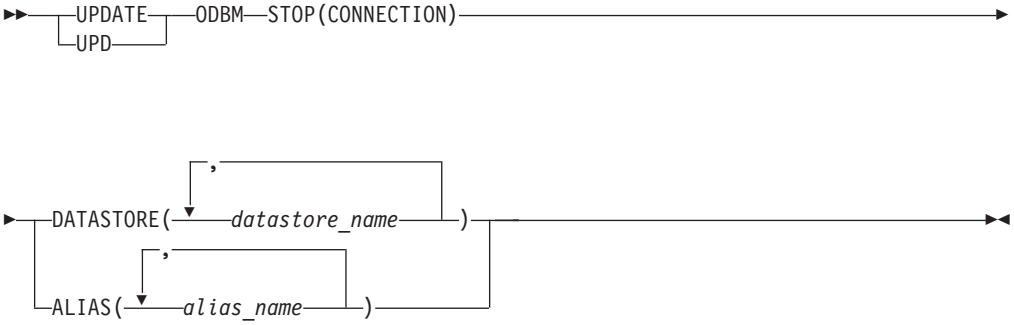
- "Environment" on page 736
- "Syntax" on page 736
- "Keywords" on page 736
- "Usage notes" on page 736
- "Output fields" on page 737
- "Return, reason, and completion codes" on page 737

• "Examples" on page 738

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the UPDATE ODBM STOP(CONNECTION) command.

ALIAS()
Specifies the 1–4 character alias name. Wildcards can be specified for the alias name. You can specify more than one aliases.

There is no default for ALIAS(). Specifying the ALIAS keyword will stop ODBM connections to all the aliases specified with *alias_name*. ALIAS() and DATASTORE() are mutually exclusive.

DATASTORE()
Specifies the 1–4 character data store name. Wildcards can be specified for the data store name. You can specify more than one data store.

There is no default for DATASTORE(). Specifying the DATASTORE keyword will stop ODBM connections to all the data stores specified with *datastore_name* as well as their associated aliases. DATASTORE() and ALIAS() are mutually exclusive.

To start connections to the data stores, you must issue an UPDATE ODBM START(CONNECTION) DATASTORE() command. Even if the actual data store subsystems are recycled after the UPDATE ODBM STOP(CONNECTION) DATASTORE() command, ODBM will not attempt to connect to the data stores until an UPDATE ODBM START(CONNECTION) DATASTORE() command is issued.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 367. Output field descriptions for the UPDATE ODBM STOP(CONNECTION) command

Short label	Keyword	Meaning
ALIAS	ALIAS	The 1–4 character alias name that is associated with an IMS data store connection.
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DSTR	N/A	Data store name.
MBR	N/A	Name of the ODBM member that processed the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE ODBM STOP(CONNECTION) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 368. Return and reason codes for the UPDATE ODBM STOP(CONNECTION) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE ODBM STOP(CONNECTION) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 369. Completion codes for the UPDATE ODBM STOP(CONNECTION) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM STOP(CONNECTION) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
6	DATASTORE DISCONNECT FAILED	An attempt to stop a data store connection failed. Message CSL4007W accompanies this completion code and explains the AIB return code, AIB reason code, and AIB error text. (See note)
10	NO RESOURCES FOUND	No resources were found.

Note: The command response is encapsulated in <rsp> and </rsp> tags. Message CSL4007W is returned within <msg> and </msg> tags.

Examples

Example 1 for UPDATE ODBM STOP(CONNECTION) command

TSO SPOC input:

```
UPD ODBM STOP(CONNECTION) DATASTORE(IMS*)
```

TSO SPOC output:

```
MbrName     DatastoreName       CC
ODBM01OD    IMS1                 0
ODBM01OD    IMS2                 0
ODBM02OD    IMS3                 0
```

Explanation: The UPDATE command will stop connections with all data stores whose name begin with “IMS”.

Example 2 for UPDATE ODBM STOP(CONNECTION) command

TSO SPOC input:

```
UPD ODBM STOP(CONNECTION) ALIAS(IO*)
```

TSO SPOC output:

```
MbrName    AliasName  CC    DatastoreName
ODBM01OD    IO1A       0     IMS1
ODBM01OD    IO1B       0     IMS1
ODBM01OD    IO2A       0     IMS2
ODBM01OD    IO2B       0     IMS2
ODBM02OD    IO3A       0     IMS3
ODBM02OD    IO3B       0     IMS3
```

Explanation: The UPDATE command will stop ODBM connections with the ODBM members whose alias names begin with “IO”.

UPDATE ODBM STOP(TRACE) command

Use the UPDATE ODBM STOP(TRACE) command to request that ODBM traces be stopped.

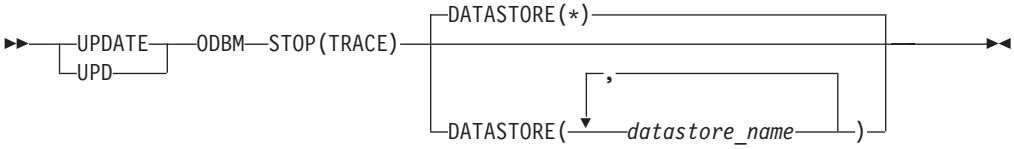
Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Output fields"
- "Return, reason, and completion codes" on page 740
- "Examples" on page 741

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the UPDATE ODBM STOP(TRACE) command.

DATASTORE()
Specifies the 1–4 character data store name. Wildcards can be specified for the data store name. The *datastore_name* is a repeatable parameter. The default is DATASTORE(*), which applies to all data store names that are known to ODBM.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 370. Output field descriptions for the UPDATE ODBM STOP(TRACE) command

Short label	Keyword	Meaning
ALIAS	N/A	Alias name.
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DSTR	N/A	Data store name.
MBR	N/A	Name of the ODBM member that processed the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE ODBM STOP(TRACE) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 371. Return and reason codes for the UPDATE ODBM STOP(TRACE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE ODBM STOP(TRACE) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 372. Completion codes for the UPDATE ODBM STOP(TRACE) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM STOP(TRACE) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for UPDATE ODBM STOP(TRACE) command

TSO SPOC input:

```
UPD ODBM STOP(TRACE) DATASTORE(*)
```

TSO SPOC output:

```
MbrName     DatastoreName  CC
ODBM010D    IMS1            0
ODBM010D    IMS2            0
ODBM020D    IMS3            0
```

Explanation: The UPDATE command will stop ODBM traces for all data stores that are known to ODBM.

UPDATE ODBM TYPE(CONFIG) command

Use the UPDATE ODBM TYPE(CONFIG) command to update active ODBM configuration.

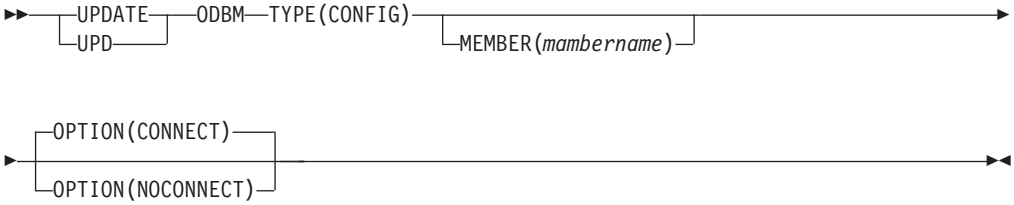
Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 742
- "Output fields" on page 742
- "Return, reason, and completion codes" on page 743
- "Examples" on page 744

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax



Keywords

The following keywords are valid for the UPDATE ODBM TYPE(CONFIG) command.

MEMBER()
Specifies a 3–character CSLDCxxx suffix that identifies which ODBM configuration PROCLIB member is used to update the active ODBM configuration. The default is the current CSLDCxxx member.

Note: If a new CSLDCxxx member is used to update the active ODBM configuration, the new member becomes the current ODBM configuration PROCLIB member. Also, when restarting ODBM, the ODBMCFG= parameter on the EXEC statement or within the CSLDIxxx initialization PROCLIB member will be used to configure ODBM.

OPTION()
Specifies whether you want connections with the data stores to be attempted after the active ODBM configuration has been updated.

CONNECT
Specifies that after updating the active ODBM configuration, all data stores within the new configuration will be connected and all aliases associated with the data stores will be made available for use. CONNECT is the default.

NOCONNECT
Specifies that after updating the active ODBM configuration, no connections will be attempted. To start connections to the data stores, you must issue an UPDATE ODBM START(CONNECTION) DATASTORE() command. Even if the actual data store subsystems are started or recycled after the UPDATE ODBM TYPE(CONFIG) OPTION(NOCONNECT) command, ODBM will not attempt to connect to the data stores until an UPDATE ODBM START(CONNECTION) DATASTORE() command is issued.

TYPE(CONFIG)
Specifies to update the active ODBM configuration.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

The UPDATE ODBM TYPE(CONFIG) command allows the user to update the active ODBM configuration by loading an updated copy of the current or an alternate CSLDCxxx ODBM configuration PROCLIB member.

Before processing the UPD ODBM TYPE(CONFIG) command, all data store connections must be stopped on all ODBMs to which the command will be routed.

Output fields

The following table shows the UPDATE ODBM output fields. The columns in the table are as follows:

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 373. Output field descriptions for the UPDATE ODBM TYPE(CONFIG) command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
CFG	N/A	ODBM configuration PROCLIB member (CSLDCxxx) used to establish the new active configuration.
MBR	N/A	IMSplex member that built the output line. The ODBM identifier of the ODBM that built the output line.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE ODBM TYPE(CONFIG) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 374. Return and reason codes for the UPDATE ODBM TYPE(CONFIG) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE ODBM TYPE(CONFIG) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005000'	CSLDALCB allocation failed.
X'04000014'	X'0000500C'	CSLDDSCB allocation failed.
X'04000014'	X'00005010'	BPEGETM failed to acquire a parse work area.
X'04000014'	X'00005028'	BPEPARSE internal error.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.
X'04000014'	X'00005088'	CSLDPRP allocation failed.
X'04000014'	X'0000508C'	BPERDPDS internal failure.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 375. Completion codes for the UPDATE ODBM TYPE(CONFIG) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM TYPE(CONFIG) command completed successfully for the resource.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
3	DATASTORE NOT AVAILABLE	An attempt to start a data store connection failed. The data store is unavailable.
4	DATASTORE CONNECTION FAILED	An attempt to start a data store connection failed. Message CSL4006W accompanies this completion code and explains the AIB return code, AIB reason code, and AIB error text.
5	DATASTORE NOT RRS CAPABLE	An attempt to start a data store connection failed. The data store is not RRS capable.
8	CONFIGURATION FILE NOT FOUND	The CSLDCxxx IMS PROCLIB member could not be located.
9	THIS ODBM NOT DEFINED	The current ODBM definition could not be located in the CSLDCxxx IMS PROCLIB member.
A	DUPLICATE ODBM DEFINED	Multiple ODBM definitions were found for the current ODBM.
B	DATASTORES NOT DEFINED	The current ODBM definition has no data stores defined.
C	DUPLICATE DATASTORES DEFINED	Duplicate data store names were found within the current ODBM definition.
D	DATASTORE NAME INVALID	A data store name within the current ODBM definition contains invalid characters.
E	DUPLICATE ALIAS DEFINED	A data store definition within the current ODBM definition contains duplicate alias names.
F	ALIAS NAME INVALID	An alias name defined within the current ODBM definition contains invalid characters.
10	NO RESOURCES FOUND	No resources were found.
11	BPEPARSE ERROR	BPEPARSE failed while parsing the CSLDCxxx IMS PROCLIB member. BPE0003E accompanies this completion code.
12	DATASTORES NOT STOPPED	Data store connections to the current ODBM have not been stopped before issuing the UPD ODBM TYPE(CONFIG) command.

Examples

Example 1 for UPDATE ODBM TYPE(CONFIG) command

TSO SPOC input:

```
UPD ODBM TYPE(CONFIG) MEMBER(009) OPTION(NOCONNECT)
```

TSO SPOC output:

```
MbrName      CC   ConfigName
ODBM01OD      0    CSLDC009
ODBM01OD      0    CSLDC009
ODBM02OD      0    CSLDC009
```

Explanation: The MEMBER(009) keyword indicates that CSLDC009 will be used to configure both ODBM01 and ODBM02. To allow this to occur, both ODBM configurations must be specified in the CSLDC009 PROCLIB member. When using the NAME() keyword, routing of this command requires caution to insure the intended results are achieved.

Example 2 for UPDATE ODBM TYPE(CONFIG) command

TSO SPOC input:

```
UPD ODBM TYPE(CONFIG)
```

TSO SPOC output:

```
MbrName      CC    ConfigName
ODBM01OD      0     CSLDC001
ODBM01OD      0     CSLDC001
ODBM02OD      0     CSLDC002
```

Explanation: This example shows that the current configuration members were, and remain to be, CSLDC001 for ODBM1 and CSLDC002 for ODBM2, respectively.

UPDATE OLREORG command

Use the UPDATE OLREORG command to change the rate and the DEL or NODEL data set or just the DEL or NODEL data set disposition flags of an owned HALDB Online Reorganization (OLR).

If you issue this command as a type-1 command, the command response is returned as a DFS0725I pre-edit message.

Subsections:

- "Environment"
- "Syntax" on page 746
- "Keywords" on page 746
- "Command responses for /UPDATE OLREORG" on page 747
- "Output fields" on page 747
- "Return, reason, and completion codes" on page 748
- "Examples" on page 749

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE OLREORG command and keywords can be issued.

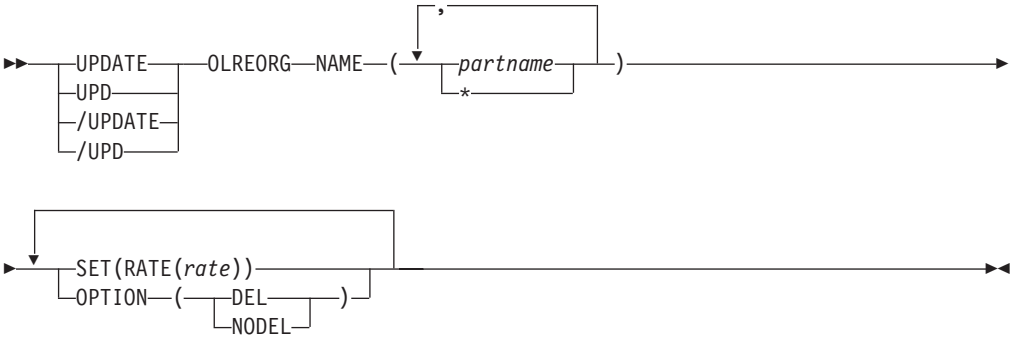
Table 376. Valid environments for the UPDATE OLREORG command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE OLREORG	X	X	
NAME	X	X	
SET	X	X	

Table 376. Valid environments for the UPDATE OLREORG command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
OPTION	X	X	

Syntax



Keywords

The following keywords are valid for the UPDATE OLREORG command:

NAME()
Specifies one or more PHDAM or PHIDAM HALDB partitions to be updated.

For the type-1 version of the command, you can specify only one partition name.

A parameter with the wildcard character (*) is not allowed, except as NAME(*) for all HALDB partitions.

SET(RATE)
Specifies the rate at which the HALDB OLR is run.

rate
You can specify a value of 1 to 100 for the rate value.

You can use the RATE parameter to control the intensity at which the reorganization runs. This can affect both the reorganization's speed and its impact on the rest of the system. The value you specify for rate is the percentage of elapsed time to be devoted to copying records. The remaining time is to be an intentionally introduced delay in the copying process that minimizes the reorganization's impact on other IMS work and on the whole system.

An online reorganization's impact on the system is affected by the available system resources, by total system utilization (including other online reorganizations), by total logging volume, by log contention, and by the intensity at which this reorganization was requested to run. These same factors also affect the speed at which the reorganization runs.

A rate value of 50 specifies that 50% of the elapsed time be spent copying records and the remaining 50% be spent in a delay. This causes the reorganization to run approximately twice as long as it would have run with a rate value of 100.

OPTION()

Allows the specification of the DEL or NODEL options.

DEL | NODEL
Specifies whether the deletion of the inactive data sets is to be attempted when the online reorganization completes. DEL and NODEL are mutually exclusive keywords.

DEL
The deletion of the inactive data sets is to be attempted when the online reorganization completes. The attempted deletion occurs regardless of who created the data sets or when the data sets were created.

NODEL
The deletion of the inactive data sets is not to be attempted when the online reorganization completes.

If the partition is tracked at an RSR tracker site, the OPTION (DEL | NODEL) value in effect at the completion of the HALDB OLR also determines whether the inactive data sets for the shadow partition are deleted at the completion of the tracking of the reorganization.

If the HALDB Online Reorganization is stopped before completion, the DEL or NODEL keyword is not remembered and will need to be specified on the INITIATE OLREORG command that is issued to resume the stopped HALDB OLR.

Command responses for /UPDATE OLREORG

When the /UPDATE OLREORG command is entered as a type-1 command, the command response is returned in a message format.

When the command completes successfully message DFS0725I is returned to the system console and master terminal with a completion code of 0. If the command results in an error, a nonzero completion code or an error message is returned to the master terminal and system console.

```
DFS0725I INITIATE|UPDATE|TERMINATE OLREORG COMMAND FOR DB dbnamexx COMPLETE.
CC=  nn
  where: dbnamexx is the HALDB partition name entered on the command
         nn       is the completion code
```

Output fields

This section describes the responses from the OM API for the UPDATE OLREORG command. The following table shows the UPDATE OLREORG output fields. The columns in the table are as follows:

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies the command keyword that caused the field to be generated.

N/A appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 377. Output fields for UPDATE OLREORG command

Short label	Keyword	Meaning
PART	N/A	Partition name.
MBR	N/A	The IMS that built the command response line.
CC	N/A	Completion code.

Return, reason, and completion codes

The OM return and reason codes that might be returned as a result of the UPDATE OLREORG command are standard for all commands that are entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 378. Return and reason codes for the UPDATE OLREORG command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE OLREORG command completed successfully.
X'00000004'	X'00001010'	No matches found for filter.
X'00000008'	X'00002008'	Insufficient number of keywords specified.
X'00000008'	X'00002040'	No filter, an invalid filter, or an insufficient number of filters was specified.
X'0000000C'	X'00003000'	At least one request was successful.
X'0000000C'	X'00003004'	None of the requests was successful.
X'00000010'	X'0000400C'	Command issued on an XRF alternate.
X'00000010'	X'00004014'	Command issued on an RSR tracker.
X'00000014'	X'00005000'	A GETMAIN error occurred.

The following table includes an explanation of the completion codes. Errors unique to the processing of UPDATE OLREORG command are returned as completion codes. A completion code is returned for each action attempted on a HALDB partition.

Table 379. Completion codes for the UPDATE OLREORG command

Completion code	Meaning
0	The UPDATE OLREORG command completed successfully for the partition.
10	Resource name is invalid.
14	Resource is not a partition name.
1C	Resource is a partitioned secondary index.
24	No HALDB OLR is in progress.
CF	Parameter value conflict, or invalid parameter value.

Examples

The following are examples of the UPDATE OLREORG command:

Example 1 for /UPDATE OLREORG

Entry ET:

```
/UPD OLREORG NAME(PDHDOKA) SET(RATE(25))
```

Response ET:

```
DFS0725I UPDATE    OLREORG COMMAND FOR DB PDHDOKA  COMPLETE. CC=   0
```

Explanation: The UPDATE OLREORG command is issued for partition PDHDOKA to change the OLR rate to 25. The command is successful as indicated in the message DSF0725 command response.

Example 2 for UPDATE OLREORG

TSO SPOC input:

```
UPD OLREORG NAME(PDHDOKA,PDHDOKB) SET(RATE(25))
```

TSO SPOC output:

```
Partition MbrName        CC
    PDHDOKA   IMSA        0
    PDHDOKA   IMS1       24
    PDHDOKB   IMSA        0
    PDHDOKB   IMS1       24
```

OM API input:

```
CMD (UPD OLREORG NAME(PDHDOKA,PDHDOKB) SET(RATE(25)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.168 21:17:57.712194</statime>
<stotime>2003.168 21:17:57.713062</stotime>
<staseq>B996297E02942007</staseq>
<stoseq>B996297E02CA6487</stoseq>
<rqsttkn1>USRT005 10141757</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003004</rsn>
<rsntext>No requests were successful</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMSA    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>OLREORG         </kwd>
```

```
<input>UPD OLREORG NAME(PDHDOKA,PDHDOKB) SET(RATE(25)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PART" llbl="Partition" scope="LCL" sort="A" key="1"
 scroll="NO" len="9" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="N" key="0" scroll="NO"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="N" key="0" scroll="YES"
 len="4" dtype="INT" align="right" />
</cmdrsphdr>
<cmdrspdata>
<rsp> PART(PDHDOKA ) MBR(IMSA    ) CC(   0) </rsp>
<rsp> PART(PDHDOKB ) MBR(IMSA    ) CC(   0) </rsp>
<rsp> PART(PDHDOKA ) MBR(IMS1    ) CC(  24) </rsp>
<rsp> PART(PDHDOKB ) MBR(IMS1    ) CC(  24) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE OLREORG command is issued for partitions PDHDOKA and PDHDOKB to update the OLR rate to 25. The command is successful at IMSA, where OLR is in progress and is not successful at IMS1, where no OLR is in progress. A completion code of 24 is returned in the IMS1 response.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

UPDATE OTMADESC command

Use the UPDATE OTMADESC command to update an existing OTMA destination descriptor without restarting IMS.

Subsections:

- "Environment"
- "Syntax" on page 751
- "Keywords" on page 751
- "Usage notes" on page 752
- "Output fields" on page 752
- "Return, reason, and completion codes" on page 753
- "Examples" on page 754

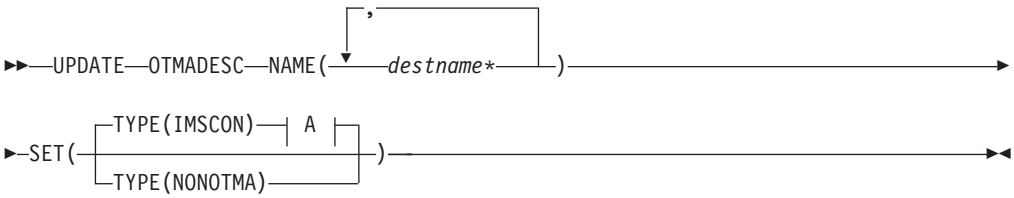
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

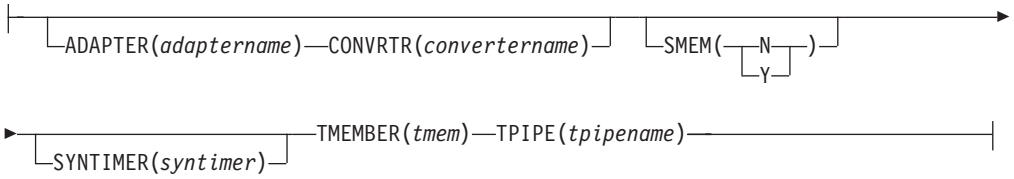
Table 380. Valid environments for the UPDATE OTMADESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE OTMADESC	X		X
NAME	X		X
SET	X		X

Syntax



A:



Keywords

The following keywords are valid for the UPDATE OTMADESC command:

NAME(*destname**)**
A required keyword whose value cannot be modified.

SET()
Specifies to update the descriptors with at least one of the following keywords. The update applies to each of the descriptor names specified in the NAME keyword. If an asterisk is appended to the descriptor name, the update will not be applied to the group of names it is trying to mask.

ADAPTER(*adaptername***)**
An optional parameter for TYPE(IMSCON) that identifies the IMS Connect adapter. *adaptername* is a 1- to 8-character adapter name. To clear the value of this keyword, specify the keyword with no value, for example, ADAPTER(). If the ADAPTER keyword is deleted, the CONVRTR keyword is also deleted. ADAPTER and TYPE(NONOTMA) are mutually exclusive.

CONVRTR(*convertername***)**
A required parameter only if the ADAPTER keyword is specified. *convertername* is a 1- to 8-character converter name used by the adapter. The value of this keyword will be automatically deleted if the value of the ADAPTER keyword is being deleted. Coding this keyword without the ADAPTER keyword or with TYPE(NONOTMA) is not valid.

SMEM(N | Y)
An optional parameter that can either be a Y or N value to indicate whether the TMEMBER name specified in the TMEMBER parameter is a super member. To clear the value of this keyword, specify the keyword with no value, for example, SMEM(). Deleting SMEM() will default to SMEM(N). If the TMEMBER name is a super member, the length of the TMEMBER name has a maximum of 4 characters. SMEM and TYPE(NONOTMA) are mutually exclusive.

SYNTIMER(*syntimer***)**
An optional parameter used for synchronous callout processing. In

synchronous callout processing, a request is canceled if an ACK/NAK or response is not received by OTMA from the client within the time specified. This value is expressed in hundredths of a second or milliseconds. The value can be numeric with a maximum length of six digits. The value has a range of 0 through 999999 and when zero is specified, it will default to 10 seconds or 1000 hundredths of a second. If you specify SYNTIMER() or SYNTIMER( ), the timeout value will be reset as if no timeout value has been specified.

TMEMBER(*tmem***)**
A required parameter for TYPE(IMSCON). If the TYPE of the descriptor is being changed from IMSCON to NONOTMA, the value of this keyword will be deleted. If the TYPE of the descriptor is being changed from NONOTMA to IMSCON, then the TMEMBER keyword must be coded. *tmem* is a 16-character OTMA TMEMBER name or a 4-character super member. TMEMBER and TYPE(NONOTMA) are mutually exclusive.

TPIPE(*tpipename***)**
An optional parameter that is a 1- to 8-character TPIPE name when TYPE(IMSCON) is specified. To delete the value of this keyword, specify the keyword with no value, for example, TPIPE(). If this keyword is not specified, the TPIPE name is the destination name specified in the NAME keyword. TPIPE and TYPE(NONOTMA) are mutually exclusive.

TYPE(IMSCON | NONOTMA)

A required keyword that can either be IMSCON or NONOTMA. This keyword determines whether the output is destined for IMS Connect (IMSCON) or non-OTMA (NONOTMA). If TYPE(IMSCON) is specified, the TMEMBER is a required keyword and the rest of the keywords, such as TPIPE, SMEM, and ADAPTER, are optional. If TYPE(NONOTMA) is specified, coding any of these optional keyword parameters is not allowed.

The destination routing descriptors can be changed from TYPE(IMSCON) to TYPE(NONOTMA) or vice versa. If the TYPE keyword is changed from IMSCON to NONOTMA, the values of the rest of the keywords (TMEMBER, TPIPE, SMEM, ADAPTER, and CONVRTR) will be deleted. If the TYPE keyword is changed from NONOTMA to IMSCON, the TMEMBER keyword must be coded.

For TYPE(IMSCON), the optional keywords can be coded with no values to delete any values for the keywords in question, for example, coding TPIPE() will clear the value of TPIPE(PIPERONE) for the descriptor.

Usage notes

The UPDATE OTMADESC command is used to modify existing destination routing descriptors that were either included in DFSYDTx or added by using the CREATE OTMADESC command. A log record will be written to track changes and to persist from one IMS restart to another. The log record will also be used to track the currency of updates to an XRF alternate and to an RSR tracking environment. The checkpoint record is X4035 and the log record for the CREATE, UPDATE, and DELETE commands is X221B.

Output fields

The following table shows the UPDATE OTMADESC output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 381. Output fields for the UPDATE OTMADESC command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" for more information. The completion code is always returned.
CCTXT	CCText	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
DEST	DestName	NAME	Destination name.
MBR	MbrName	N/A	Member name.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the UPDATE OTMADESC command. The OM return and reason codes that might be returned as a result of the UPDATE OTMADESC command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 382. Return and reason codes for the UPDATE OTMADESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See Table 383 on page 754 for details.
X'02000008'	X'00002000'	The command contains an invalid verb or no client is registered for the verb.
X'02000008'	X'00002004'	The command contains an invalid primary keyword or no client is registered for the keyword.
X'02000008'	X'00002028'	The command contains an invalid keyword.
X'02000008'	X'0000202C'	The command contains an unknown positional parameter.
X'02000008'	X'00002034'	The command contains an incomplete keyword parameter.
X'02000008'	X'00002038'	The command is missing a required parameter.

Table 382. Return and reason codes for the UPDATE OTMADESC command (continued)

Return code	Reason code	Meaning
X'02000008'	X'0000203C'	The command contains an invalid keyword parameter value.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 383. Completion codes for the UPDATE OTMADESC command

Completion code	Completion code text	Meaning
0	Command completed successfully	The UPDATE OTMADESC command completed successfully for the resource.
151	Descriptor not found	The descriptor name does not exist.
153	Adapter blank, Convrtr not blank	The value of the ADAPTER keyword is blank and the value of the CONVRTR keyword is not blank. Either both keywords must be blank or both keywords must contain a valid name.
154	Adapter not blank, Convtr blank	The value of the ADAPTER keyword is not blank and the value of the CONVRTR keyword is not blank. Either both keywords must be blank or both keywords must contain a valid name.
156	SMEM(Y), super mbr name > 4 char	The super member name in the TMEMBER keyword must have a maximum length of 4 characters.
159	TMEMBER is required for IMSCON	The type is being changed from NONOTMA to IMSCON, but the value of TMEMBER is blank.
162	Descriptor not available for upd	During update processing, the descriptor was deleted by another user.
167	SYNTIMER must have numeric value	The timeout value must have a value expressed in numbers within parenthesis.
169	SYNTIMER has nonnumeric value	The value must not contain alphabetic characters or any character that is not numeric.

Examples

The following are examples of the UPDATE OTMADESC command:

Example 1 for UPDATE OTMADESC command

TSO SPOC input:

```
UPDATE OTMADESC NAME(OTMACL*,OTMACL99) SET(TYPE(NONOTMA))
```

TSO SPOC output:

```
DestName   MbrName     CC
OTMACL99   IMSA         0
OTMACL*    IMSA         0
```

Explanation: This UPDATE command changes the output type to NONOTMA. This keyword will delete any other keywords (such as TMEM or TPIPE), because no other keywords are relevant for TYPE(NONOTMA).

Example 2 for UPDATE OTMADESC command

TSO SPOC input:

```
UPDATE OTMADESC NAME(OTMACL*,OTMACL99) SET(TPIPE(HWS1TP02))
```

TSO SPOC output:

```
DestName   MbrName     CC
OTMACL99   IMSA         0
OTMACL*    IMSA         0
```

Explanation: This UPDATE command changes the TPIPE name to HWS1TP02. The TPIPE name will be changed from HWS1TP01 to HWS1TP02 for OTMACL99 and the TPIPE for OTMACL* will be set to HWS1TP02.

Example 3 for UPDATE OTMADESC command

TSO SPOC input:

```
UPDATE OTMADESC NAME(OTMACL*) SET(TPIPE())
```

TSO SPOC output:

```
DestName  MbrName     CC
OTMACL*   IMSA         0
```

Explanation: This UPDATE command for descriptor OTMACL* will delete the TPIPE name by coding TPIPE(). An asterisk in the UPDATE command does not update the group of names the asterisk is masking. It updates only the OTMACL* entry in the table of destination routing descriptors. It does not update OTMACL99.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

UPDATE PGM command

Use the UPDATE PGM command to update program resources.

A program resource defines the resource requirements for application programs that run under the control of the DB/TM environment, as well as for application programs that access databases through DBCTL. Program resources combined with transactions define the scheduling and resource requirements for an application program.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 757
- "Usage notes" on page 762
- "Output fields" on page 763
- "Return, reason, and completion codes" on page 763
- "Examples" on page 767

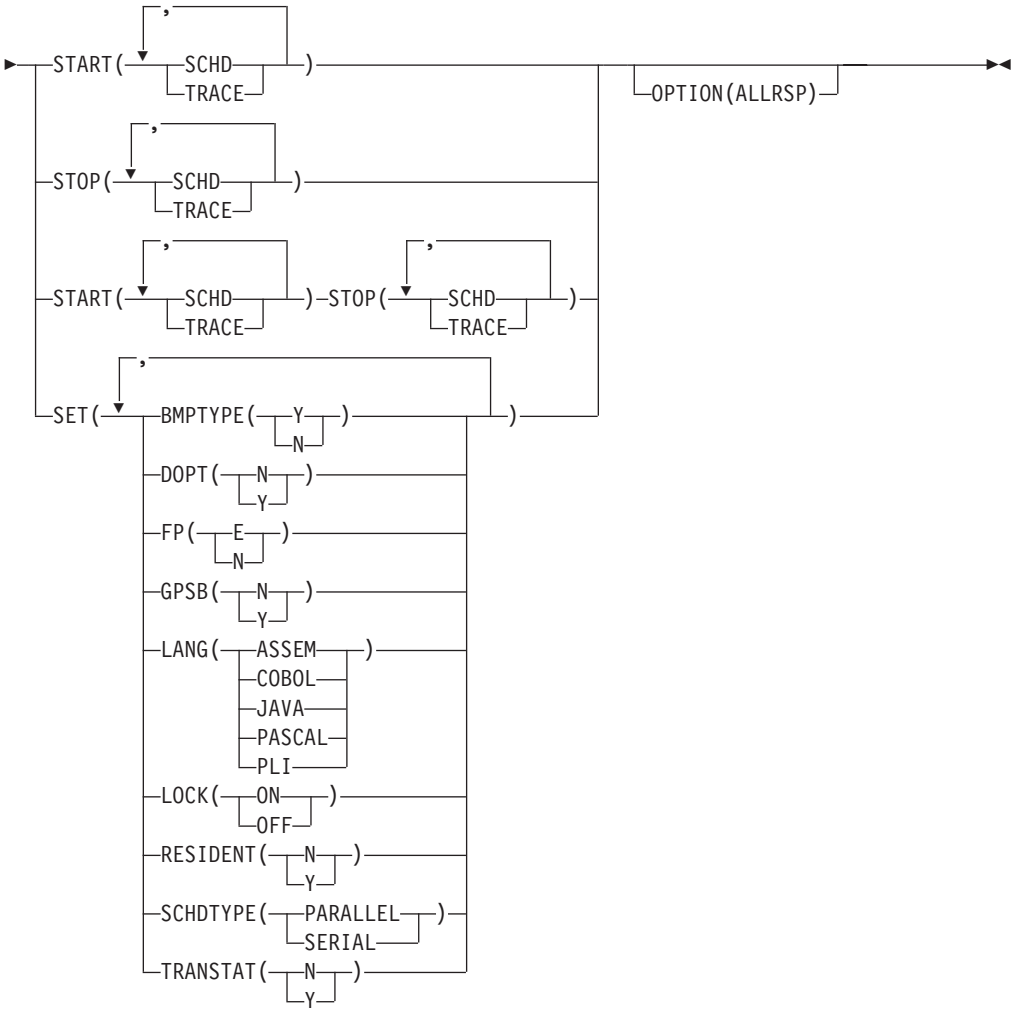
Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 384. Valid environments for the UPDATE PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE PGM	X	X	X
NAME	X	X	X
OPTION	X	X	X
SET	X	X	X
START	X	X	X
STOP	X	X	X

Syntax



Keywords

The following keywords are valid for the UPDATE PGM command:

NAME
Specifies the 1-8 character name of the program. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the resource names that are processed. For NAME(*), command responses are returned only for the resource names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the resource names that are processed.

OPTION
Specifies additional functions to be performed along with the command.

ALLRSP
Indicates that the response lines are to be returned for all resources that are processed on the command. The default action is to return response lines only for the resources that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SET
Specifies the attribute values to be changed.

BMPTYPE
BMP type option. Specifies whether the program runs in a BMP type region (Y) or not (N). A BMP type region can be a BMP region or a JBP region. Updating this attribute stops the program from scheduling during command processing.

PSBs scheduled by DB2 stored procedures, by programs running under WebSphere Application Server, and by other users of the ODBA interface might be defined with BMPTYPE Y or N.

Y The program runs in a BMP type region. It might run in an IMS BMP region or a JBP region. Any associated transactions are assigned normal and limit priority values of zero.

N The program does not run in a BMP type region. It might run in an IMS TM MPP, JMP, or IFP region or it might use either the ODBA or DRA interface. This specification should be used for programs running in IMS TM MPP, JMP, and IFP regions, or PSBs scheduled by CICS programs using DBCTL and other users of the DRA interface. This is the default.

Keyword combination rules include the following:

- FP(E) and BMPTYPE(Y) are mutually exclusive.

DOPT
Specifies the dynamic option.

N The PSB associated with this program is not located dynamically. The PSB must exist in ACBLIB, otherwise the program is set to a NOTINIT status, and cannot be scheduled.

Y The PSB associated with this program is located dynamically. Each time the program associated with this PSB is scheduled, the latest copy of the PSB is loaded from ACBLIB. The PSB does not need to be in any data set defined for ACBLIB until it is actually required to process a transaction. A new version of the PSB can be genned in ACBLIB and is picked up the next time the PSB is scheduled. DOPT PSBs referencing DBDs that are missing from ACBLIB cannot be scheduled. When the program terminates, the PSB is deleted from the PSB pool.

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:

- DOPT(Y) and GPSB(Y) are mutually exclusive.
- DOPT(Y) and LANG(JAVA) is a valid combination.
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- RESIDENT(Y) and DOPT(Y) are mutually exclusive.
- SCHDTYPE(PARALLEL) and DOPT(Y) are mutually exclusive.

FP Specifies the Fast Path option.

E The program is a Fast Path-exclusive application program. This implicitly defines a wait-for-input (WFI) application program. Either a transaction or a routing code that can be assigned by the user Input Edit/Routing exit routine must be defined for the Fast Path-exclusive application, in order for this program to be usable.

N The program is not a Fast Path-exclusive application program. When FP(N) is specified, any attempt to use Fast Path resources or commands might yield unpredictable results.

Updating this attribute is rejected if any routing codes or transactions that reference this program have conflicting attributes. To update this attribute, you might need to delete the routing codes and transactions that reference it. Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:

- FP(E) requires Fast Path to be defined.
- LANG(JAVA) and FP(E) are mutually exclusive.
- BMPTYPE(Y) and FP(E) are mutually exclusive.

GPSB

Specifies the generated PSB option.

N The PSB associated with the program is not generated by IMS. The PSB must exist in ACBLIB, otherwise the program is set to a NOTINIT status, and cannot be scheduled.

Y The PSB associated with the program is generated by IMS. It is not loaded from ACBLIB. The scheduling process of all environments generates a PSB containing an I/O PCB and an alternate modifiable PCB. You do not need to perform the PSBGEN and ACBGEN, thus eliminating I/O to the ACBLIB. The generated PSB contains an I/O PCB named IOPCBbbb and a modifiable, alternate PCB named TPPCB1bb. With an alternate modifiable PCB, an application can use the CHNG call to change the output destination and send output to a destination other than the input destination.

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:

- DOPT(Y) and GPSB(Y) are mutually exclusive.
- DOPT(Y) and LANG(JAVA) is a valid combination.
- GPSB(Y) requires LANG.
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- RESIDENT(Y) and GPSB(Y) are mutually exclusive.

LANG

Specifies the language interface of the program for a GPSB, or defines a DOPT(Y) program as using the Java language.

In order to define a DOPT program using the Java language, the program must be defined with DOPT(Y) and LANG(JAVA). DOPT PSBs are not loaded at IMS restart; they are loaded every time the program is scheduled. When the program is scheduled for the first time, IMS does not know the language until after the program is scheduled in a region and the PSB is loaded. Unless LANG(JAVA) is defined for the DOPT(Y) program, the program is incorrectly scheduled in a non-Java region.

The LANG parameters and their meanings are identified in the following table.

LANG parameter	Meaning
ASSEM	Assembly language
COBOL	COBOL
JAVA	Java
PASCAL	Pascal
PLI	PL/I

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:

- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- DOPT(Y) and LANG(JAVA) is a valid combination.
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- FP(E) and LANG(JAVA) are mutually exclusive.

LOCK

Specifies that the LOCK status is to be set on or off. SET(LOCK(ON | OFF)) cannot be specified with any other SET attribute. SET(LOCK(ON | OFF)) can be specified with the START or STOP keyword.

ON Locks the program and prevents it from being scheduled.

OFF

Unlocks the program and allows it to be scheduled.

RESIDENT

Specifies the resident option. The RESIDENT(N) option takes effect immediately. The RESIDENT(Y) option takes effect at the next restart, unless an error is encountered such as no PSB in the ACBLIB for the program, or if the program was updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart.

N The PSB associated with the program is made to be not resident in storage. The PSB is loaded at scheduling time.

Y The PSB associated with the program is made resident in storage at IMS cold start or restart. IMS loads the PSB and initializes it. A resident PSB is accessed in local storage, which avoids I/O to the ACBLIB.

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:

- DOPT(Y) and RESIDENT(Y) are mutually exclusive.
- GPSB(Y) and RESIDENT(Y) are mutually exclusive.

SCHDTYPE

Specifies whether this application program can be scheduled into more than one message region or batch message region simultaneously.

PARALLEL

The application program can be scheduled in multiple regions simultaneously.

SERIAL
The application program can be scheduled in only one region at a time.

Updating this attribute to SCHDTYPE(SERIAL) is rejected if a transaction that references this program is defined with a parallel limit count other than 65535.

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:

- DOPT(Y) and SCHDTYPE(PARALLEL) are mutually exclusive.

TRANSTAT
Specifies whether transaction level statistics should be logged. The value specified has meaning only if the program is a JBP or a non-message driven BMP. If Y is specified, transaction level statistics are written to the log in an X'56FA' log record.

N Transaction level statistics should not be logged.

Y Transaction level statistics should be logged.

The TRANSTAT keyword on the UPDATE PGM or UPDATE PGMDESC command gives the user the ability to override the system default or the current value of the TRANSTAT parameter. If the TRANSTAT keyword is omitted on the UPDATE PGM command, the current transaction level statistics setting is unchanged for the program.

START
Specifies attributes that are to be started.

SCHD
Starts scheduling of the application program. Or, resets the NOTINIT status if set because of an invalid database or DMB, in case the database or DMB has been created since the scheduling failure. If the database or DMB is still invalid when the program is next scheduled, the program will be marked with a status of NOTINIT again.

TRACE

Starts the tracing of the DL/I portion of Data Communications (DC) for the application program. Each DL/I call to a TPPCB, issued by the user application program, is traced on entry to and exit from the DC call handler DFSDLA30. On entry to DFSDLA30 a type 6701-LA3A record is written, on exit from DFSDLA30 a type 6701-LA3B record is written.

Each record contains the following items if applicable:

- TPPCB
- Up to 64 bytes of the I/O area
- SMB
- PST

If the batch message program (BMP) being traced is the IBM IMS Queue Control Facility for z/OS (5697-E99), a 6701-MRQB record is logged by the IMS Queue Control Facility module DFSQMRQ0. The default program name for the IMS Queue Control Facility BMP is MRQPSB, and can be overridden on the MSGQUEUE system definition macro.

Items logged in the 6701-MRQB record, if applicable, are:

- TPPCB
- AIB
- I/O area
- PST
- QTPDST
- QSAPWKAD
- QMBA
- PSTDCA
- REG14-12

STOP
Specifies attributes that are to be stopped.

SCHD
Stops scheduling of the application program.

TRACE
Stops the tracing of the DL/I portion of Data Communications (DC) for the application program.

Usage notes

Resources exist for the life of the IMS unless they are deleted by using a DELETE command. Resources are recoverable across an IMS warm start or emergency restart. Resources are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE PGM command can be issued only through the OM API. This command applies to DB/DC, DBCTL and DCCTL systems. This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE PGM command specified with SET() is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined), except for SET(LOCK()) and SET(TRANSTAT()). This command is recoverable.

The UPDATE commands are not allowed for IMS-defined resources, except with keywords START, STOP, SET(LOCK(ON|OFF)), and SET(TRANSTAT(Y|N)). For the UPDATE PGM command, this means Fast Path utility program DBF#FPU0.

The UPDATE PGM command changes a MODBLKS program to a dynamic program with a definition type of UPDATE, if the BMPTYPE, DOPT, FP, GPSB, LANG, RESIDENT, SCHDTYPE, or TRANSTAT value is changed.

Each program is updated individually, unlike the online change process where either all programs are updated or no programs are updated. Most runtime resource definition values for a program can be updated only if the program is not in use. If the program is in use, the update fails.

You can update the status of a program (LOCK, START, STOP) while the program is in use.

If all the attributes specified by the UPDATE command are already defined for the resource, no update is actually made, no resources are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary processor usage when no action needs to be taken.

The following program attributes cannot be updated if online change for MODBLKS is enabled: BMPTYPE, DOPT, FP, GPSB, LANG, RESIDENT, SCHDTYPE.

Output fields

The following table shows the UPDATE PGM output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 385. Output fields for the UPDATE PGM command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code
CCTXT	CCText	*error*	Completion code text that briefly explains the nonzero completion code.
ERRT	ErrorText	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
MBR	MbrName	N/A	IMSplex member that built the output line.
PGM	PgmName	PGM	Program name.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 386. Return and reason codes for the UPDATE PGM command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000004'	X'00002008'	Invalid number of keywords. Either a SET, START, or STOP keyword is required.

Table 386. Return and reason codes for the UPDATE PGM command (continued)

Return code	Reason code	Meaning
X'00000008'	X'00002040'	More than one filter value is specified on the UPDATE PGM command. Confirm that only one of SET or START | STOP filters are specified on the command.
X'00000008'	X'00002044'	The UPDATE PGM command is not processed because the same attribute value was specified for the START and STOP filters. The attributes SCHD and TRACE can be specified only with the START or STOP keyword, not both. Confirm that only one START | STOP attribute is specified on the command.
X'00000008'	X'00002048'	Invalid SET attribute.
X'00000008'	X'00002106'	DOPT(Y) is mutually exclusive with RESIDENT(Y).
X'00000008'	X'00002107'	DOPT(Y) is mutually exclusive with SCHDTYPE(PARALLEL).
X'00000008'	X'0000210B'	FP(E) is mutually exclusive with BMPTYPE(Y).
X'00000008'	X'0000210D'	FP(E) is mutually exclusive with LANG(JAVA).
X'00000008'	X'00002113'	GPSB(Y) is mutually exclusive with DOPT(Y).
X'00000008'	X'00002114'	GPSB(N) is mutually exclusive with LANG.
X'00000008'	X'00002115'	GPSB(Y) is mutually exclusive with RESIDENT(Y).
X'00000008'	X'00002132'	DOPT(Y) not supported with LANG except for LANG(JAVA).
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), only resources with nonzero completion codes are returned. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined, FP(E) is invalid.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 387. Completion codes for the UPDATE PGM command

Completion code	Completion code text	Meaning	Error text
0		Command completed successfully for program.	
10	NO RESOURCES FOUND	Program name is invalid, or the wildcard parameter specified does not match any program names.	
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this program. This could also mean this command, if the program is specified by more than one specific or wildcard parameter.	
29	DOPT(Y)/RESIDENT(Y) CONFLICT	Program update failed because dynamic DOPT(Y) option conflicts with resident RESIDENT(Y) option.	
2A	DOPT(Y)/ PARALLEL CONFLICT	Program update failed because dynamic DOPT(Y) option conflicts with parallel schedule SCHDTYPE(PARALLEL) option.	
2F	FP(E)/ BMPTYPE(Y) CONFLICT	Program update failed because Fast Path exclusive FP(E) option conflicts with BMP type BMPTYPE(Y).	
36	FP(E)/FP(N) PGM CONFLICT	Program update failed because non-Fast Path option FP(N) conflicts with Fast Path exclusive transaction that references this program.	
37	FP(E)/LANG(JAVA) CONFLICT	Program update failed because Fast Path exclusive FP(E) option conflicts with Java language LANG(JAVA).	
3E	FP(N)/FP(E) PGM CONFLICT	Program update failed because Fast Path exclusive FP(E) option conflicts with non-Fast Path transaction that references this program.	*tranname* (8 chars)
3F	FP(P)/ BMPTYPE(Y) CONFLICT	Program update failed because BMP type program conflicts with Fast Path potential transaction that references this program.	*tranname* (8 chars)

Table 387. Completion codes for the UPDATE PGM command (continued)

Completion code	Completion code text	Meaning	Error text
40	PARLIM/ SCHDTYPE(SERIAL) CONFLICT	The SCHDTYPE value cannot be changed to SERIAL because a transaction that references the program has a PARLIM value that is something other than 65535. The program definition is not updated.	
43	GPSB(Y)/DOPT(Y) CONFLICT	Program updated failed because generated PSB GPSB(Y) option conflicts with dynamic DOPT(Y) option.	
46	GPSB(N)/LANG CONFLICT	Generated PSB option N (GPSB(N)) conflicts with the language option (LANG()). The program definition is not updated.	
47	GPSB(Y)/RESIDENT(Y) CONFLICT	Program update failed because generated PSB GPSB(Y) option conflicts with the resident RESIDENT(Y) option.	
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for the IMS-defined resource. DBF#FPU0 is an example of an IMS-defined program.	
73	PROGRAM SCHEDULED	Program is scheduled.	
7A	RTC/FP(N) PGM CONFLICT	Program to be updated as FP(N) conflicts with a routing that references the program. The routing code name is returned as error text. Suggested actions: Issue DELETE RTC command or DELETE TRAN (if associated with FP exclusive transaction) to delete the routing code.Or Issue UPDATE RTC command to update the program to another name.	*rtcodename* (8 chars)

Table 387. Completion codes for the UPDATE PGM command (continued)

Completion code	Completion code text	Meaning	Error text
97	DOPT(Y)/LANG CONFLICT	Program update failed because dynamic option DOPT(Y) conflicts with LANG specified. DOPT(Y) is only supported with LANG(JAVA).	
99	NOT INITIALIZED	Update program failed because the program was not successfully initialized. You can use the QRY PGM SHOW(STATUS) command to return additional information regarding the reason for the NOTINIT status.	
B9	REQUIRES LANG	Program update failed because the DOPT(Y) or GPSB(Y) option specified requires language LANG to be specified or already defined.	
BA	NOT ALLOWED FOR MRQ PSB	UPDATE command failed for MRQ PSB because an attribute was specified that is not supported for the MRQ PSB: BMPTYPE(N), DOPT(Y), GPSB(Y), FP(E), RESIDENT(Y), SCHDTYPE(SERIAL).	

Examples

The following are examples of the UPDATE PGM command:

Example 1 for UPDATE PGM command

TSO SPOC input:

```
UPDATE PGM NAME(BADNAME,AUTPSB2,CDEBS,BMP011,BAD*) SET(SCHDTYPE(PARALLEL))
```

TSO SPOC output:

```
Response for: UPDATE PGM NAME(BADNAME,AUTPSB2,CDEBS,BMP011,BAD*) SET(SCHDTYPE(PARALLEL))
PgmName  MbrName    CC CCText
AUTPSB2  IMS1        0
BAD*     IMS1       10 NO RESOURCES FOUND
BADNAME  IMS1       10 NO RESOURCES FOUND
BMP011   IMS1       2A DOPT(Y)/PARALLEL CONFLICT
CDEBS    IMS1        0
```

OM API input:

```
CMD(UPDATE PGM NAME(BADNAME,AUTPSB2,CDEBS,BMP011,BAD*) SET(SCHDTYPE(PARALLEL)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.311 00:55:21.981467</statime>
<stotime>2006.311 00:55:21.981970</stotime>
<staseq>BFAADF0EA2A1B548</staseq>
<stoseq>BFAADF0EA2C12E48</stoseq>
<rqsttkn1>USRT011 10165521</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>PGM             </kwd>
<input>UPDATE PGM NAME(BADNAME,AUTPSB2,CDEBS,BMP011,BAD*)
SET(SCHDTYPE(PARALLEL)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PGM" llbl="PgmName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PGM(BADNAME ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>PGM(AUTPSB2 ) MBR(IMS1) CC(   0) </rsp>
<rsp>PGM(CDEBS   ) MBR(IMS1) CC(   0) </rsp>
<rsp>PGM(BMP011  ) MBR(IMS1) CC(  2A) CCTXT(DOPT(Y)/PARALLEL CONFLICT)
</rsp>
<rsp>PGM(BAD*    ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update some programs so that they can be scheduled in multiple regions simultaneously. The update completed successfully for programs AUTPSB2 and CDEBS, as shown by the completion code 0. The update failed for program BMP011 with completion code 2A, indicating that the SCHDTYPE(PARALLEL) attribute conflicts with the DOPT(Y) attribute already defined for program BMP011. The update fails for program BADNAME and for parameter BAD* with completion code 10, since program BADNAME does not exist and no program name starts with BAD.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Diagnosing problems in the Queue Control Facility Message Requeuer (Diagnosis)

Related tasks:

Updating runtime application program resource and descriptor definitions with the UPDATE command (System Definition)

Related reference:

/LOCK command (Commands)

Command keywords and their synonyms (Commands)

"/START PGM command" on page 485

"/STOP PGM command" on page 532

Chapter 27, "/TRACE commands," on page 585

"/UNLOCK PGM command" on page 623

"/TRACE PGM command" on page 603

IMS Queue Control Facility overview

UPDATE PGMDESC command

Use the UPDATE PGMDESC command to update program descriptors. A descriptor is a model that can be used to create descriptors or resources.

Updating a descriptor changes only the attributes explicitly specified on the UPDATE command. Attributes not specified retain their existing values. Any program resource or descriptor can be created using this descriptor as a model, by specifying the CREATE LIKE(DESC(*descriptor_name*)) command. Any descriptor or resource that was already created using this descriptor is not updated.

Subsections:

- "Environment"
- "Syntax" on page 770
- "Keywords" on page 770
- "Usage notes" on page 774
- "Output fields" on page 774
- "Return, reason, and completion codes" on page 775
- "Examples" on page 777

Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which you can use the commands and keywords.

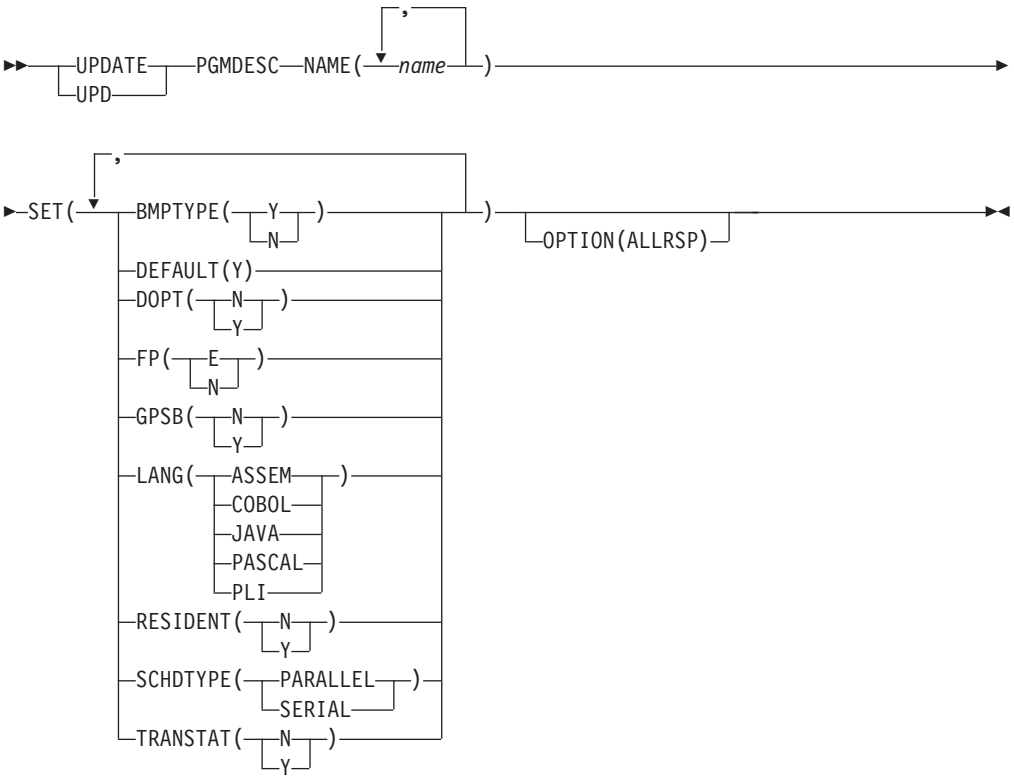
Table 388. Valid environments for the UPDATE PGMDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE PGMDESC	X	X	X
NAME	X	X	X
OPTION	X	X	X

Table 388. Valid environments for the UPDATE PGMDESC command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
SET	X	X	X

Syntax



Keywords

The following keywords are valid for the UPDATE PGMDESC command:

NAME
Specifies the 1-8 character name of the program descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the descriptor names that are processed. For NAME(*), command responses are returned only for the descriptor names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the descriptor names that are processed.

OPTION
Specifies additional functions to be performed along with the command.

ALLRSP
Indicates that the response lines are to be returned for all descriptors that are processed on the command. The default action is to return response lines only for the descriptors that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SET

Specifies the attribute values to be changed.

BMPTYPE

BMP type option. Specifies whether the program executes in a BMP type region or not. A BMP type region can be a BMP region or a JBP region.

PSBs scheduled by DB2 stored procedures, by programs running under WebSphere Application Server, and by other users of the ODBA interface may be defined with BMPTYPE Y or N.

Y The program executes in a BMP type region. It can execute in an IMS BMP region or a JBP region. Any associated transactions are assigned normal and limit priority values of zero.

N The program does not execute in a BMP type region. It may execute in an IMS TM MPP, JMP, or IFP region, or it may use either the ODBA or DRA interface. This specification should be used for programs running in IMS TM MPP, JMP, and IFP regions, or PSBs scheduled by CICS programs using DBCTL and other users of the DRA interface. This is the default.

Keyword combination rules include the following:

- BMPTYPE(Y) and FP(E) are mutually exclusive.

DEFAULT(Y)

Specifies this descriptor as the default, which resets the existing default descriptor to DEFAULT(N). When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a program descriptor called DFSDSPG1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor. Since only one program descriptor can be the default at one time, only one descriptor name can be specified with DEFAULT(Y).

DOPT

Specifies the dynamic option.

N The PSB associated with this program is not located dynamically. The PSB must exist in ACBLIB, otherwise the program is set to a NOTINIT status, and cannot be scheduled.

Y The PSB associated with this program is located dynamically. Each time the program associated with this PSB is scheduled, the latest copy of the PSB is loaded from ACBLIB. The PSB does not need to be in any data set defined for ACBLIB until it is actually required to process a transaction. A new version of the PSB can be genned in ACBLIB and is picked up the next time the PSB is scheduled. DOPT PSBs referencing DBDs that are missing from ACBLIB cannot be scheduled. When the program terminates, the PSB is deleted from the PSB pool.

Keyword combination rules include the following:

- DOPT(Y) and GPSB(Y) are mutually exclusive.
- DOPT(Y) and LANG(JAVA) is a valid combination.
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.

- RESIDENT(Y) and DOPT(Y) are mutually exclusive.
- SCHDTYPE(PARALLEL) and DOPT(Y) are mutually exclusive.

FP Specifies the Fast Path option.

E The program is a Fast Path-exclusive application program. This implicitly defines a wait-for-input (WFI) application program. Either a transaction or a routing code that can be assigned by the user Input Edit/Routing exit routine must be defined for the Fast Path-exclusive application, in order for this program to be usable.

N The program is not a Fast Path-exclusive application program. When FP(N) is specified, any attempt to use Fast Path resources or commands will yield unpredictable results.

Keyword combination rules include the following:

- FP(E) requires Fast Path to be defined.
- LANG(JAVA) and FP(E) are mutually exclusive.
- BMPTYPE(Y) and FP(E) are mutually exclusive.

GPSB

Specifies the generated PSB option.

N The PSB associated with the program is not generated by IMS. The PSB must exist in ACBLIB; otherwise the program is set to a NOTINIT status and cannot be scheduled.

Y The PSB associated with the program is generated by IMS. It is not loaded from ACBLIB. The scheduling process of all environments generates a PSB containing an I/O PCB and an alternate modifiable PCB. You do not need to perform the PSBGEN and ACBGEN, thus eliminating I/O to the ACBLIB. The generated PSB contains an I/O PCB named IOPCBbbb and a modifiable, alternate PCB named TPPCB1bb. With an alternate modifiable PCB, an application can use the CHNG call to change the output destination and send output to a destination other than the input destination.

Keyword combination rules include the following:

- DOPT(Y) and GPSB(Y) are mutually exclusive.
- DOPT(Y) and LANG(JAVA) is a valid combination.
- GPSB(Y) requires LANG.
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- RESIDENT(Y) and GPSB(Y) are mutually exclusive.

LANG

Specifies the language interface of the application program for a GPSB.

The LANG parameters and their meanings are identified in the following table.

LANG parameter	Meaning
ASSEM	Assembler
COBOL	COBOL
JAVA	Java
PASCAL	Pascal

LANG parameter	Meaning
PLI	PL/I

Keyword combination rules include the following:

- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- DOPT(Y) and LANG(JAVA) is a valid combination.
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- FP(E) and LANG(JAVA) are mutually exclusive.

RESIDENT
Specifies the resident option.

N The PSB associated with the program is made to be not resident in storage. The PSB is loaded at scheduling time.

Y The PSB associated with the program is made to be resident in storage immediately. IMS loads the PSB and initializes it. The PSB is removed from the PSB pool, if applicable. A resident PSB is accessed in local storage, which eliminates I/O to the ACBLIB.

Keyword combination rules include the following:

- DOPT(Y) and RESIDENT(Y) are mutually exclusive.
- GPSB(Y) and RESIDENT(Y) are mutually exclusive.

SCHDTYPE

Specifies whether this application program can be scheduled into more than one message region or batch message region simultaneously.

PARALLEL
The application program can be scheduled in multiple regions simultaneously.

SERIAL
The application program can be scheduled in only one region at a time.

Updating this attribute to SCHDTYPE(SERIAL) is rejected if a transaction referencing this program is defined with a parallel limit count other than 65535.

Keyword combination rules include the following:

- DOPT(Y) and SCHDTYPE(PARALLEL) are mutually exclusive.

TRANSTAT
Specifies whether transaction level statistics should be logged. The value specified has meaning only if the program is a JBP or a non-message driven BMP. If Y is specified, transaction level statistics are written to the log in a X'56FA' log record.

N Transaction level statistics should not be logged.

Y Transaction level statistics should be logged.

The TRANSTAT keyword on the UPDATE PGMDESC command gives the user the ability to override the system default or the current value of the TRANSTAT parameter. If the TRANSTAT keyword is omitted on the UPDATE PGMDESC command, the current transaction level statistics setting is unchanged for the program.

Usage notes

If all the attributes specified by the UPDATE command are already defined for the descriptor, no update is actually made, no descriptors are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

Descriptors exist for the life of the IMS unless they are deleted using a DELETE command. Descriptors are recoverable across an IMS warm start or emergency restart. Descriptors are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE PGMDESC command can be issued only through the OM API. This command applies to DB/DC, DBCTL and DCCTL systems. This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE PGMDESC commands are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined). This command is recoverable.

If the descriptor is the IMS-defined program descriptor (DFSDSPG1) the only attributes that can be updated are the DEFAULT (SET(DEFAULT(Y))) and the TRANSTAT (SET(TRANSTAT(Y|N|))) attributes.

Each descriptor is updated individually. Individual updating does not work like online change where either all descriptors are updated or no descriptors are updated.

Output fields

The following table shows the UPDATE PGMDESC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 389. Output fields for the UPDATE PGMDESC command

Short label	Keyword	Meaning
CC	N/A	Completion code
CCTXT	*error*	Completion code text that briefly explains the nonzero completion code.
DESC	PGMDESC	Program descriptor name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
MBR	N/A	IMSplex member that built the output line.

Table 389. Output fields for the UPDATE PGMDESC command (continued)

Short label	Keyword	Meaning
OLDDEF	PGMDESC	Old default descriptor name, if this descriptor is updated to be the default by specifying DEFAULT(Y). The old default descriptor is no longer the default.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 390. Return and reason codes for the UPDATE PGMDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000004'	X'00002008'	Invalid number of keywords. A SET keyword is required.
X'00000008'	X'00002048'	Invalid SET attribute.
X'00000008'	X'00002106'	DOPT(Y) mutually exclusive with RESIDENT(Y).
X'00000008'	X'00002107'	DOPT(Y) mutually exclusive with SCHDTYPE(PARALLEL).
X'00000008'	X'0000210B'	FP(E) mutually exclusive with BMPTYPE(Y).
X'00000008'	X'0000210D'	FP(E) mutually exclusive with LANG(JAVA).
X'00000008'	X'00002113'	GPSB(Y) mutually exclusive with DOPT(Y).
X'00000008'	X'00002114'	GPSB(N) is mutually exclusive with LANG.
X'00000008'	X'00002115'	GPSB(Y) mutually exclusive with RESIDENT(Y).
X'00000008'	X'00002132'	DOPT(Y) not supported with LANG except for LANG(JAVA).
X'00000008'	X'00002133'	Multiple name parameters were specified with DEFAULT(Y). Only one descriptor can be the default at a time.
X'0000000C'	X'00003000'	Command was successful for some descriptors but failed for others. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), only descriptors with non-zero completion codes are returned. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined, FP(E) is invalid.

Table 390. Return and reason codes for the UPDATE PGMDESC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 391. Completion codes for the UPDATE PGMDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for program descriptor.
10	NO RESOURCES FOUND	Program descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this program descriptor. This could also mean this command, if the descriptor is specified by more than one specific or wildcard parameter.
29	DOPT(Y)/RESIDENT(Y) CONFLICT	Program descriptor update failed because dynamic DOPT(Y) option conflicts with resident RESIDENT(Y) option.
2A	DOPT(Y)/ PARALLEL CONFLICT	Program descriptor update failed because dynamic DOPT(Y) option conflicts with parallel schedule SCHDTYPE(PARALLEL) option.
2F	FP(E)/ BMPTYPE(Y) CONFLICT	Program descriptor update failed because Fast Path exclusive FP(E) option conflicts with BMP type BMPTYPE(Y).
37	FP(E)/LANG(JAVA) CONFLICT	Program descriptor update failed because Fast Path exclusive FP(E) option conflicts with Java language LANG(JAVA).
40	PARLIM/ SCHDTYPE(SERIAL) CONFLICT	The SCHDTYPE value cannot be changed to SERIAL because a transaction that references the program has a PARLIM value that is something other than 65535. The program definition is not updated.
43	GPSB(Y)/DOPT(Y) CONFLICT	Program descriptor updated failed because generated PSB GPSB(Y) option conflicts with dynamic DOPT(Y) option.
46	GPSB(N)/LANG CONFLICT	Generated PSB option N (GPSB(N)) conflicts with the language option (LANG()). The program definition is not updated.

Table 391. Completion codes for the UPDATE PGMDESC command (continued)

Completion code	Completion code text	Meaning
47	GPSB(Y)/RESIDENT(Y) CONFLICT	Program descriptor update failed because generated PSB GPSB(Y) option conflicts with the resident RESIDENT(Y) option.
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for the IMS descriptor or resource. DFSDSPG1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
8A	WILDCARD PARAMETER NOT SUPPORTED	A wildcard parameter was specified with DEFAULT(Y). Only one descriptor can be the default at a time.
97	DOPT(Y)/LANG CONFLICT	Program descriptor update failed because dynamic option DOPT(Y) conflicts with LANG specified. DOPT(Y) is only supported with LANG(JAVA).

Examples

The following are examples of the UPDATE PGMDESC command:

Example 1 for UPDATE PGMDESC command

TSO SPOC input:

```
UPDATE PGMDESC NAME(*) SET(BMPTYPE(Y)) OPTION(ALLRSP)
```

TSO SPOC output:

```
Response for: UPDATE PGMDESC NAME(*) SET(BMPTYPE(Y)) OPTION(ALLRSP)
DescName MbrName    CC CCText
DESC001  IMS1        0
DESC002  IMS1        0
DESC003  IMS1        0
DESC004  IMS1        0
DESC005  IMS1        0
DFSDSPG1 IMS1       48 NOT ALLOWED FOR IMS RESOURCE
DOPTDESC IMS1        0
FPEDESC  IMS1       2F FP(E)/BMPTYPE(Y) CONFLICT
GPSBDESC IMS1        0
PARLDESC IMS1        0
RESDESC  IMS1        0
TLSDESC  IMS1        0
```

OM API input:

```
CMD(UPDATE PGMDESC NAME(*) SET(BMPTYPE(Y)) OPTION(ALLRSP))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.312 23:30:46.559300</statime>
<stotime>2006.312 23:30:46.559923</stotime>
<staseq>BFAD4FE141E44D08</staseq>
<stoseq>BFAD4FE1420B3F48</stoseq>
```

```
<rqsttkn1>USRT011 10153046</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>PGMDESC         </kwd>
<input>UPDATE PGMDESC NAME(*) SET(BMPTYPE(Y)) OPTION(ALLRSP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DESC004 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC005 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC001 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DOPTDESC) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(RESDESC ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DFSDSPG1) MBR(IMS1) CC(  48) CCTXT(NOT ALLOWED FOR IMS
 RESOURCE) </rsp>
<rsp>DESC(TLSDESC ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC002 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(PARLDESC) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC003 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1) CC(  2F) CCTXT(FP(E)/BMPTYPE(Y) CONFLICT)
 </rsp>
<rsp>DESC(GPSBDESC) MBR(IMS1) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update all program descriptors to be BMPTYPE(Y). The update completed successfully for most program descriptors, as shown by completion code 0. The update failed for IMS-defined descriptor DFSDSPG1 with completion code 48, because the only attribute that can be updated for DFSDSPG1 is DEFAULT(Y). The update failed for program descriptor FPEDESC with completion code 2F, because the BMPTYPE(Y) attribute conflicts with the Fast Path exclusive FP(E) attribute already defined for program descriptor FPEDESC.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

UPDATE POOL command

Use the UPDATE POOL command to specify values associated with storage used by the Fast Path 64-bit buffer manager.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes for UPDATE POOL TYPE(FPBP64)" on page 780
- "Output fields" on page 780
- "Return, reason, and completion codes" on page 781
- "Examples" on page 781

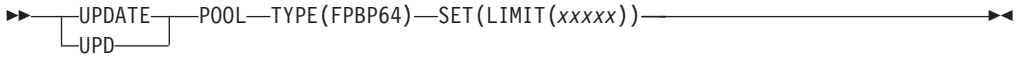
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 392. Valid environments for the UPDATE POOL command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE POOL	X	X	X
TYPE	X	X	X
SET	X	X	X

Syntax



Keywords

The following keywords are valid for the UPDATE POOL command:

TYPE()
This required keyword specifies which type of UPDATE POOL command to issue.

FPBP64
This option specifies that the command sets or changes values for storage used by the Fast Path 64-bit buffer manager. TYPE(FPBP64) is mutually exclusive with other options on the TYPE() keyword. The command is rejected with a syntax error if multiple options are specified.

SET()
Specifies the attribute values to be changed. This keyword is valid only if TYPE(FPBP64) is specified.

LIMIT
Specifies the maximum amount of 64-bit storage used by the Fast Path 64-bit buffer manager.

xxxxx
You can specify the value in bytes, kilobytes, or megabytes, as follows:

- *xxxxx* - Value in bytes. The maximum value is 2,147,483,647 bytes.
- *xxxxx*K - Value in kilobytes. The maximum value is 2,097,151 KB.
- *xxxxx*M - Value in megabytes. The maximum value is 2,047 MB.

The valid range is between 2GB-1 megabytes and the initial size of the buffer pool at IMS startup.

You can also set this value at IMS startup by specifying the parameter FPBP64M on the DFSDFxxx member of the IMS PROCLIB data set.

Usage notes for UPDATE POOL TYPE(FPBP64)

The value that can be changed by the UPDATE POOL TYPE(FPBP64) SET(LIMIT(*xxxxx*)) command is not recoverable.

Enter the UPDATE POOL TYPE(FPBP64) SET(LIMIT(*xxxxx*)) command after an IMS restart, if needed. Because this command updates the value set by FPBP64M in the DFSDFxxx member, you might want to update FPBP64M with a new value before IMS cold restart.

Output fields

The following table shows the UPDATE POOL output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

SHOW parameter
Identifies the parameter on the SHOW keyword that caused the field to be generated. N/A appears for output fields that are always returned. *Error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 393. Output fields for the UPDATE POOL command

Short label	Long label	SHOW parameter	Meaning
CC	CC	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned.
CCTXT	CCText	*Error*	Completion code text that briefly explains the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	The IMSplex member that built the output line. The IMS identifier of the IMS for which the database information is displayed. The IMS identifier is always returned.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 394. Return and reason codes for the UPDATE POOL command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'00000010'	X'0000400C'	The command failed, because it is not valid on an XRF alternate IMS.
X'00000010'	X'00004014'	The command failed, because it is not valid on an RSR tracking IMS.
X'00000010'	X'00004024'	The command failed, because FP=N is in the startup parameter.
X'00000010'	X'00004404'	The command failed, because FPBP64=N is in the startup parameter.
X'00000010'	X'0000440C'	A smaller storage usage limit than what is currently in use was specified on the command. The storage usage limit is not changed.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 395. Completion codes for the UPDATE POOL command

Completion code	Completion code text	Meaning
0		The command completed successfully.
B		A smaller storage usage limit than what is currently in use was specified on the UPDATE POOL TYPE(FPBP64) SET(LIMIT(*xxxxx*)) command. The storage usage limit is not changed.

Examples

The following are examples of the UPDATE POOL command:

Example 1 for UPDATE POOL command

TSO SPOC input:

```
UPD POOL TYPE(FPBP64) SET(LIMIT(1000M))
```

TSO SPOC output:

```
Pool Type        MbrName    CC
FPBP64           SYS3        0
```

OM API input:

```
CMD(UPD POOL TYPE(FPBP64) SET(LIMIT(1000M)))
```

OM API output:

```
<imsout>
 <ctl>
 <omname>OM1OM   </omname>
 <omvsn>1.4.0</omvsn>
 <xmlvsn>20  </xmlvsn>
 <statime>2008.318 21:54:49.340430</statime>
 <stotime>2008.318 21:54:49.340724</stotime>
 <staseq>C34A9721FEE0EC72</staseq>
 <stoseq>C34A9721FEF34C72</stoseq>
 <rqsttkn1>USRT004 10135449</rqsttkn1>
 <rc>00000000</rc>
 <rsn>00000000</rsn>
 </ctl>
 <cmd>
 <master>SYS3    </master>
 <userid>USRT004 </userid>
 <verb>UPD </verb>
 <kwd>POOL            </kwd>
 <input>UPD POOL TYPE(FPBP64) SET(LIMIT(1000M)) </input>
 </cmd>
 <cmdrsphdr>
 <hdr slbl="BTYP" llbl="Pool Type" scope="LCL" sort="a" key="1"
  scroll="no" len="8" dtype="CHAR" align="left" />
 <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
  len="4" dtype="CHAR" align="right" />
 <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
  scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
 </cmdrsphdr>
 <cmdrspdata>
 <rsp>btyp(FPBP64) MBR(SYS3    ) CC(   0)</rsp>
 </cmdrspdata>
</imsout>
```

Explanation: The buffer usage limit for the Fast Path 64-bit buffer manager was changed to 1000M (1 GB) and the command ended successfully.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

UPDATE RTC command

Use the UPDATE RTC command to update Fast Path routing codes. A Fast Path routing code can be used by the Fast Path Input Edit/Routing Exit Routine (DBFHAGU0) to route a transaction to a different application program for processing.

Subsections:

- "Environment" on page 783
- "Syntax" on page 783
- "Keywords" on page 783
- "Usage notes" on page 784
- "Output fields" on page 785
- "Return, reason, and completion codes" on page 785

• "Examples" on page 787

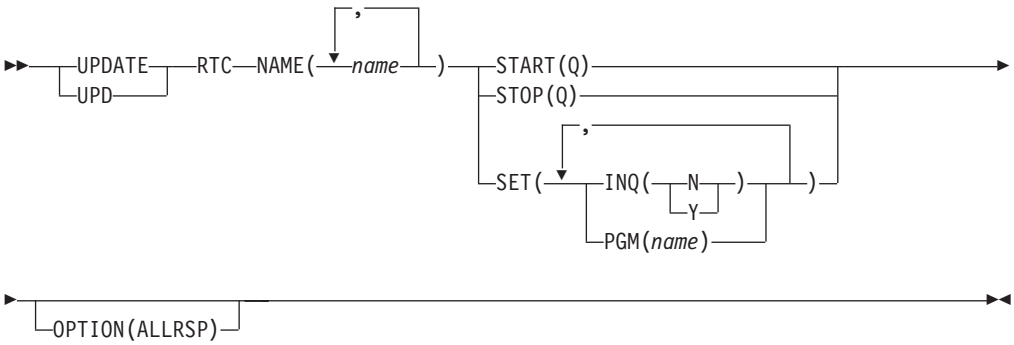
Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 396. Valid environments for the UPDATE RTC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE RTC	X		X
NAME	X		X
OPTION	X		X
SET	X		X
START	X		X
STOP	X		X

Syntax



Keywords

The following keywords are valid for the UPDATE RTC command:

NAME
Specifies the 1-8 character name of the routing code. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the resource names that are processed. For NAME(*), command responses are returned only for the resource names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the resource names that are processed.

OPTION
Specifies additional functions to be performed along with the command.

ALLRSP
Indicates that the response lines are to be returned for all resources that are processed on the command. The default action is to return response lines only for the resources that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SET
Specifies the attribute values to be changed.

INQ
Specifies the inquiry option.

N This is not an inquiry routing code.

Y This is an inquiry routing code. Any message associated with the routing code is an inquiry transaction. This option should be specified only for programs that do not cause a change to a database. Programs are prohibited from issuing Insert, Delete, or Replace calls to a database when processing an inquiry transaction.

Updating this attribute quiesces the routing code for the duration of command processing.

PGM

The name of the application program associated with the routing code. The program must exist and be defined with a BMPTYPE of N.

Updating this attribute quiesces the routing code for the duration of command processing.

START
Specifies attributes that are to be started.

Q Starts queuing of messages.

STOP
Specifies attributes that are to be stopped.

Q Stops queuing of messages.

Usage notes

Resources exist for the life of the IMS unless they are deleted using a DELETE command. Resources are recoverable across an IMS warm start or emergency restart. Resources are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE RTC command can be issued only through the OM API. Fast Path must be installed on the system. This command applies to DB/DC and DCCTL systems. This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE RTC commands specified with SET() are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined). This command is recoverable.

The UPDATE RTC command changes a MODBLKS routing code to dynamic, if the INQ or PGM attribute is changed.

Each routing code is updated individually, unlike the online change process where either all routing codes are updated or no routing codes are updated. The runtime resource definition for a routing code can be updated only if the routing code is not in use. If the routing code is in use, the update fails. An exception to this rule is status. You can update the status of a routing code while it is in use. In a sysplex environment, the update might succeed on some IMS systems and fail on others. In order to maximize the likelihood that the update will succeed, perform the following steps before attempting the update:

- Check for work in progress with a QRY RTC SHOW(WORK) command and either wait for the work to finish or address the work in progress. See the output fields for the QUERY RTC command for examples of work that might cause the delete to fail. Examples of work include a command in progress for the routing code or the routing code is active.
- The routing code cannot be updated if the program is scheduled. If the program is scheduled, you must stop the region before you issue the UPDATE RTC command.

Runtime resource definition attributes include the following: INQ and PGM. If all the attributes specified by the UPDATE command are already defined for the resource, no update is actually made, no resources are quiesced, no log record is created, and a completion code of zero is returned.

The following routing code attributes cannot be updated if online change for MODBLKS is enabled: INQ, PGM.

Output fields

The following table shows the UPDATE RTC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 397. Output fields for the UPDATE RTC command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the non-zero completion code.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a non-zero completion code and contains information that further explains the completion code.
MBR	N/A	IMSplex member that built the output line.
RTC	RTC	Routing code name.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 398. Return and reason codes for the UPDATE RTC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000008'	X'00002008'	Required keywords were not specified.
X'00000008'	X'00002123'	Invalid program name.
X'00000008'	X'00002048'	Invalid SET attribute.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), only resources with non-zero completion codes are returned. See the completion code table for details.
X'00000010'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'0000431C'	Program is quiesced. Cannot quiesce program.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 399. Completion codes for the UPDATE RTC command

Completion code	Completion code text	Meaning
0		Command completed successfully for routing code.
10	NO RESOURCES FOUND	Routing code name is invalid, or the wildcard parameter specified does not match any resource names.

Table 399. Completion codes for the UPDATE RTC command (continued)

Completion code	Completion code text	Meaning
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this routing code. This could also mean this command, if the resource is specified by more than one specific or wildcard parameter. Or, the routing code is updating the program name and another command is in progress for the program.
39	FP(E) TRAN FOR RTC EXISTS	The routing code was created by IMS for a Fast Path exclusive transaction and cannot be updated with the UPDATE RTC command. The attributes of the routing code can be updated only with the UPDATE TRAN command for the Fast Path exclusive transaction.
61	DFSBCB STORAGE ERROR	DFSBCB storage error. Could not get storage for RCTE control block.
73	PROGRAM SCHEDULED	Program is scheduled.
7A	RTC/FP(N) PGM CONFLICT	Program specified is not Fast Path exclusive.
90	INTERNAL ERROR	Internal error.
99	NOT INITIALIZED	Routing code update failed because the routing code is not initialized. QUERY RTC STATUS(NOTINIT) displays the reason why the routing code is not initialized, for example, the associated program is not defined. Correct the definition error and issue the UPD RTC START(Q) command to initialize the routing code.

Examples

The following are examples of the UPDATE RTC command:

Example 1 for UPDATE RTC command

TSO SPOC input:

```
UPD  RTC NAME(BADNAME,FPTRN02,SMQFP7*,BAD*) SET(PGM(DBFSAMP3))
```

TSO SPOC output:

```
Response for: UPD  RTC NAME(BADNAME,FPTRN02,SMQFP7*,BAD*) SET(PGM(DBFSAMP3))
RtcName  MbrName    CC CCText
BAD*     IMS1       10 NO RESOURCES FOUND
BADNAME  IMS1       10 NO RESOURCES FOUND
FPTRN02  IMS1        0
SMQFP7A  IMS1        0
SMQFP7B  IMS1        0
```

```
SMQFP7C  IMS1        0
SMQFP71  IMS1        0
SMQFP72  IMS1        0
SMQFP73  IMS1        0
```

OM API input:

```
CMD(UPDATE RTC NAME(BADNAME,FPTRN02,SMQFP7*,BAD*) SET(PGM(DBFSAMP3)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.311 23:50:02.663055</statime>
<stotime>2006.311 23:50:02.663790</stotime>
<staseq>BFAC125257C8FB80</staseq>
<stoseq>BFAC125257F6E500</stoseq>
<rqsttkn1>USRT011 10155002</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>RTC             </kwd>
<input>UPD  RTC NAME(BADNAME,FPTRN02,SMQFP7*,BAD*) SET(PGM(DBFSAMP3))
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="RTC" llbl="RtcName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RTC(BADNAME ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>RTC(FPTRN02 ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(SMQFP7A ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(BAD*    ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>RTC(SMQFP7B ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(SMQFP7C ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(SMQFP71 ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(SMQFP72 ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(SMQFP73 ) MBR(IMS1) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE RTC command is issued to update the program for several routing codes. The update completed successfully for most of the routing codes, as shown by completion code 0. The update fails for routing code BADNAME and for parameter BAD* with completion code 10 because routing code BADNAME does not exist and no routing code name starts with BAD.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

"/START RTC command" on page 489

"/STOP RTC command" on page 541

UPDATE RTCDESC command

Use the UPDATE RTCDESC command to update Fast Path routing code descriptors.

A descriptor is a model that can be used to create descriptors or resources. Updating a descriptor changes only the attributes explicitly specified on the UPDATE command. Attributes not specified retain their existing values. Any routing code resource or descriptor can be created using this descriptor as a model, by specifying the CREATE LIKE(DESC(*descriptor_name*)) command. Any descriptor or resource that was already created using this descriptor is not updated.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 790
- "Usage notes" on page 791
- "Output fields" on page 791
- "Return, reason, and completion codes" on page 792
- "Examples" on page 793

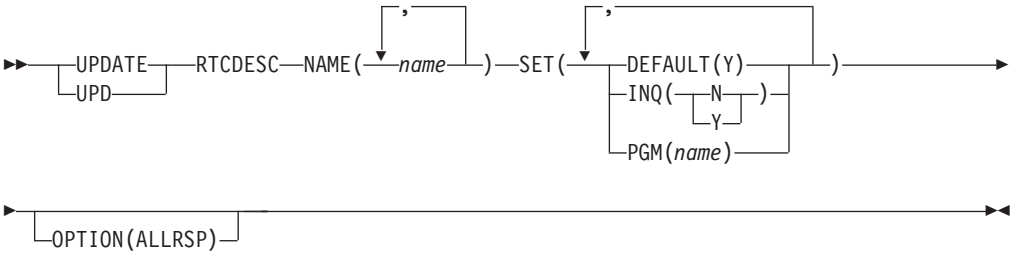
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 400. Valid environments for the UPDATE RTCDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE RTCDESC	X		X
NAME	X		X
OPTION	X		X
SET	X		X

Syntax



Keywords

The following keywords are valid for the UPDATE RTCDESC command:

NAME
Specifies the 1-8 character name of the routing code descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the descriptor names that are processed. For NAME(*), command responses are returned only for the descriptor names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the descriptor names that are processed.

OPTION
Specifies additional functions to be performed along with the command.

ALLRSP
Indicates that the response lines are to be returned for all descriptors that are processed on the command. The default action is to return response lines only for the descriptors that resulted in an error. It is valid with NAME(*) only. ALLRSP is ignored for other NAME values.

SET
Specifies the attribute values to be changed.

DEFAULT(Y)
Specifies this descriptor as the default, which resets the existing default descriptor to DEFAULT(N). When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a routing code descriptor called DBFDSRT1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor. Because only one routing code descriptor can be the default at one time, only one routing code descriptor name can be specified with DEFAULT(Y).

INQ
Specifies the inquiry option.

N This is not an inquiry routing code.

Y This is an inquiry routing code. Any message associated with the routing code is an inquiry transaction. This should be specified only for programs that do not cause a change to a database. Programs are prohibited from issuing Insert, Delete, or Replace calls to a database when processing an inquiry transaction.

PGM

The name of the application program associated with the routing code. The program must exist and be defined with a BMPTYPE of N.

Usage notes

If all the attributes specified by the UPDATE command are already defined for the descriptor, no update is actually made, no descriptors are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

Descriptors exist for the life of the IMS unless they are deleted using a DELETE command. The descriptors are recoverable across an IMS warm start or emergency restart. Descriptors are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE RTCDESC command can be issued only through the OM API. Fast Path must be installed on the system. This command applies to DB/DC and DCCTL systems. This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE RTCDESC commands are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined). These commands are recoverable.

If the descriptor is the IMS-defined routing code descriptor (DBFDSRT1), the only attribute that can be updated is the DEFAULT attribute.

Each descriptor is updated individually. Individual updating does not work like online change where either all descriptors are updated or no descriptors are updated.

Output fields

The following table shows the UPDATE RTCDESC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 401. Output fields for the UPDATE RTCDESC command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the nonzero completion code.
DESC	RTCDESC	Routing code descriptor name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.

Table 401. Output fields for the UPDATE RTCDESC command (continued)

Short label	Keyword	Meaning
MBR	N/A	IMSplex member that built the output line.
OLDDEF	RTCDESC	Old default descriptor name, if this descriptor is updated to be the default by specifying DEFAULT(Y). The old default descriptor is no longer the default.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 402. Return and reason codes for the UPDATE RTCDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000008'	X'00002008'	Required keywords were not specified.
X'00000008'	X'00002123'	Invalid program name.
X'00000008'	X'00002048'	Invalid SET attribute.
X'00000008'	X'00002133'	Multiple name parameters were specified with DEFAULT(Y). Only one descriptor can be the default at one time.
X'0000000C'	X'00003000'	Command was successful for some descriptors but failed for others. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), only descriptors with nonzero completion codes are returned. See the completion code table for details.
X'00000010'	X'00003004'	Command was not successful for any of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'0000431C'	Program is quiesced. Cannot quiesce program.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 403. Completion codes for the UPDATE RTCDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for routing code descriptor.
10	NO RESOURCES FOUND	Routing code descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this routing code descriptor. This could also mean this command, if the resource is specified by more than one specific or wildcard parameter.
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for the IMS descriptor or resource. DBFDSRT1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
61	DFSBCB STORAGE ERROR	DFSBCB storage error. Could not get storage for RCTE control block.
7A	RTC/FP(N) PGM CONFLICT	Program specified is not Fast Path exclusive.
8A	WILDCARD PARAMETER NOT SUPPORTED	A wildcard parameter was specified with DEFAULT(Y). Only one descriptor can be the default at one time.
90	INTERNAL ERROR	Internal error.

Examples

The following are examples of the UPDATE RTCDESC command:

Example 1 for UPDATE RTCDESC command

TSO SPOC input:

```
UPD RTCDESC NAME(*) SET(PGM(BMPFPE01)) OPTION(ALLRSP)
```

TSO SPOC output:

```
Response for: UPD RTCDESC NAME(*) SET(PGM(BMPFPE01)) OPTION(ALLRSP)
DescName MbrName    CC CCText
DBFDSRT1 IMS1       7A RTC/FP(N) PGM CONFLICT
DESC001  IMS1       7A RTC/FP(N) PGM CONFLICT
DESC002  IMS1       7A RTC/FP(N) PGM CONFLICT
DESC003  IMS1       7A RTC/FP(N) PGM CONFLICT
DESC004  IMS1       7A RTC/FP(N) PGM CONFLICT
DESC005  IMS1       7A RTC/FP(N) PGM CONFLICT
RTCDESC1 IMS1       7A RTC/FP(N) PGM CONFLICT
```

OM API input:

```
CMD(UPDATE RTCDESC NAME(*) SET(PGM(BMPFPE01)) OPTION(ALLRSP))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.312 00:04:22.046943</statime>
<stotime>2006.312 00:04:22.047492</stotime>
<staseq>BFAC1585EA4DF64A</staseq>
<stoseq>BFAC1585EA70488A</stoseq>
<rqsttkn1>USRT011 10160422</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003004</rsn>
<rsntxt>No requests were successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>RTCDESC         </kwd>
<input>UPD RTCDESC NAME(*) SET(PGM(BMPFPE01)) OPTION(ALLRSP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DESC004 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP(N) PGM CONFLICT)
</rsp>
<rsp>DESC(DESC005 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP(N) PGM CONFLICT)
</rsp>
<rsp>DESC(DESC001 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP(N) PGM CONFLICT)
</rsp>
<rsp>DESC(RTCDESC1) MBR(IMS1) CC(  7A) CCTXT(RTC/FP(N) PGM CONFLICT)
</rsp>
<rsp>DESC(DBFDSRT1) MBR(IMS1) CC(  7A) CCTXT(RTC/FP(N) PGM CONFLICT)
</rsp>
<rsp>DESC(DESC002 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP(N) PGM CONFLICT)
</rsp>
<rsp>DESC(DESC003 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP(N) PGM CONFLICT)
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE RTCDESC command is issued to update all routing code descriptors to reference program BMPFPE01. The update fails for all routing

codes with completion code 7A because a routing code cannot be associated with a non-Fast Path program.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

UPDATE TRAN command

Use the UPDATE TRAN command to update transaction resources.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 798
- "Usage notes" on page 817
- "Equivalent IMS type-1 commands" on page 818
- "Output fields" on page 819
- "Return, reason, and completion codes" on page 820
- "Examples" on page 826

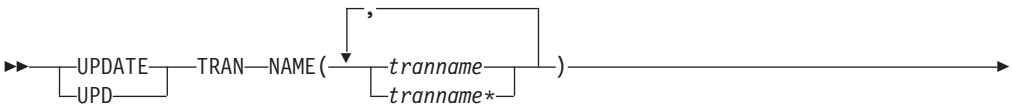
Environment

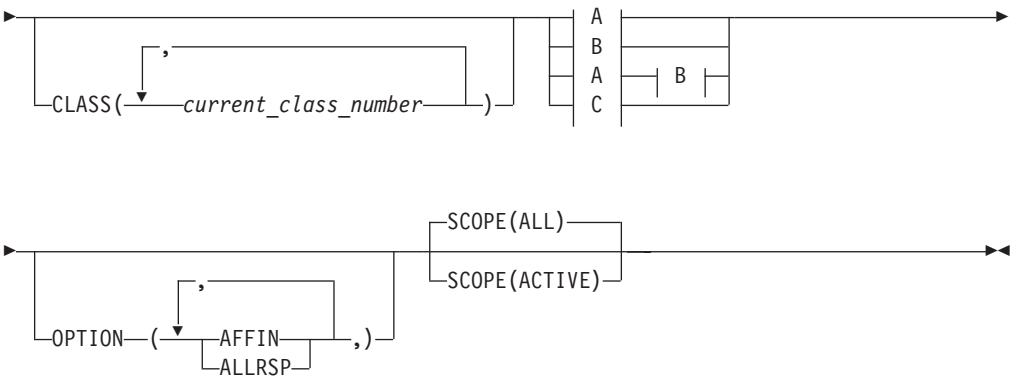
The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE TRAN command and keywords can be issued.

Table 404. Valid environments for the UPDATE TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE TRAN	X		X
CLASS	X		X
NAME	X		X
OPTION	X		X
SCOPE	X		X
SET	X		X
START	X		X
STOP	X		X

Syntax




CLASS(
,
current_class_number
)
A
B
A
B
C
SCOPE(ALL)
SCOPE(ACTIVE)
OPTION
(
,
AFFIN
ALLRSP
,)

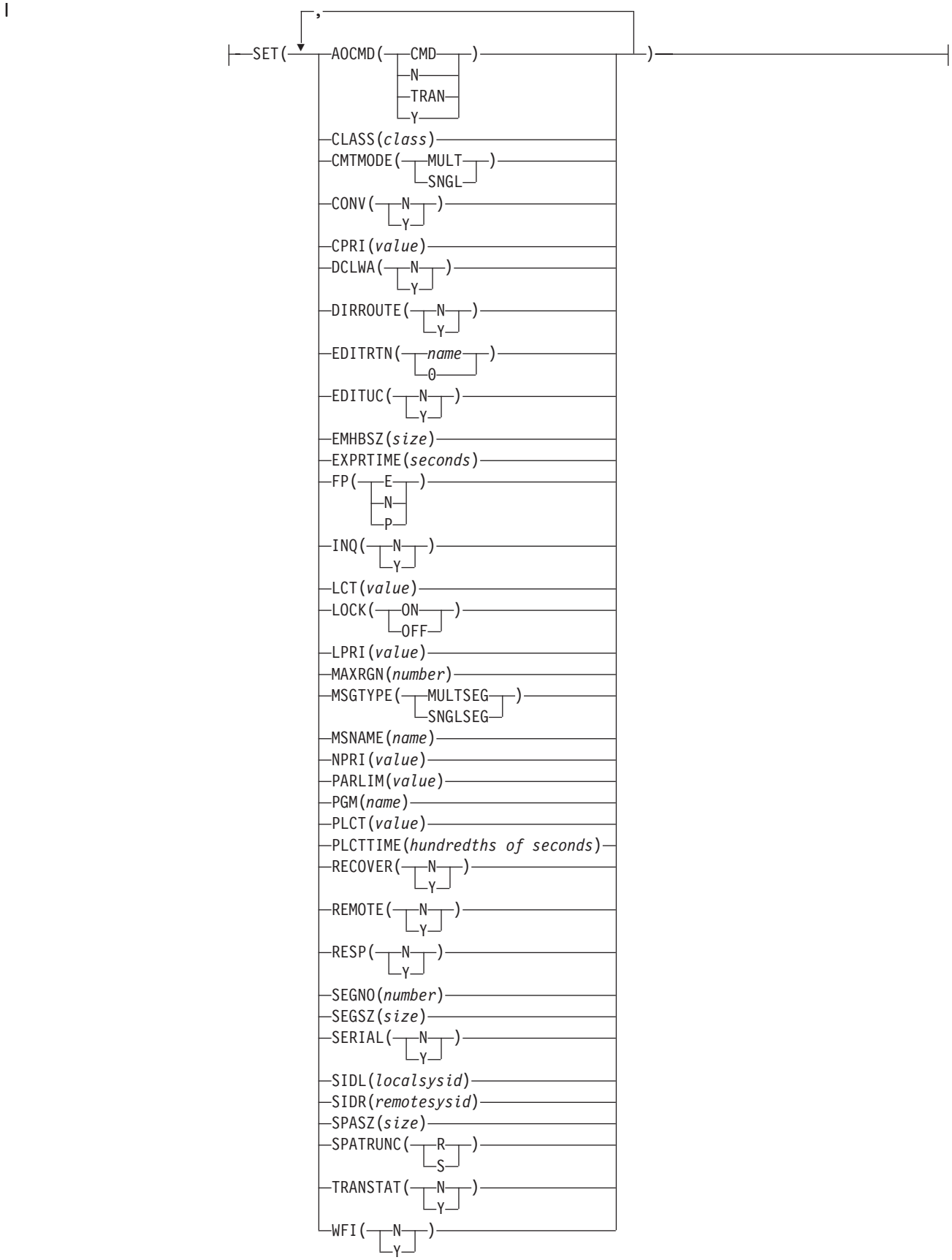
A:

START(
,
Q
SCHD
SUSPEND
TRACE
)

B:

STOP(
,
Q
SCHD
TRACE
)

C:


SET(
AOCMD(CMD|N|TRAN|Y)
CLASS(class)
CMTMODE(MULT|SNGL)
CONV(N|Y)
CPRI(value)
DCLWA(N|Y)
DIRROUTE(N|Y)
EDITRTN(name|0)
EDITUC(N|Y)
EMHBSZ(size)
EXPRTIME(seconds)
FP(E|N|P)
INQ(N|Y)
LCT(value)
LOCK(ON|OFF)
LPRI(value)
MAXRGN(number)
MSGTYPE(MULTSEG|SNGLSEG)
MSNAME(name)
NPRI(value)
PARLIM(value)
PGM(name)
PLCT(value)
PLCTTIME(hundredths of seconds)
RECOVER(N|Y)
REMOTE(N|Y)
RESP(N|Y)
SEGNO(number)
SEGSZ(size)
SERIAL(N|Y)
SIDL(localsysid)
SIDR(remotesysid)
SPASZ(size)
SPATRUNC(R|S)
TRANSTAT(N|Y)
WFI(N|Y)
,
)

Keywords

The following keywords are valid for the UPDATE TRAN command:

CLASS()
Selects the transactions associated with the specified class or classes to be updated.

NAME(*tranname*)
Specifies the 1-8 character name of the transaction. Wildcards can be specified for the *tranname*. The *tranname* is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the resource names that are processed. For NAME(*) command responses are returned only for the resource names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the resource names that are processed.

OPTION()
Specifies the additional functions to be performed. Following is a list of additional functions:

AFFIN
AFFIN is valid with START(SCHD) or STOP(SCHD).

When used with START(SCHD), OPTION(AFFIN) indicates that the command should issue an inform for the transaction with affinity to the local IMSID. This option is used to start a local affinity transaction queue in a shared-queues environment.

When used with STOP(SCHD), OPTION(AFFIN) indicates that the command should unregister a transaction for local affinity processing.

The UPDATE TRAN START(SCHD) OPTION(AFFIN) command does not support generic transaction names or ALL in the NAME() filter.

You can use the UPDATE TRAN NAME(*tranname*) START(SCHD) OPTION(AFFIN) command to start a local affinity transaction queue in a shared-queues environment. Starting a local affinity transaction queue in a shared-queues environment causes this IMS to be notified when transactions with an affinity name of this IMSs IMSID or RSENAME (XRF) appended to the transaction name, are on the shared queues for processing. This affinity registration is in addition to the normal transaction registration (that is, registration with the transaction name appended with blanks). When a transaction is started with affinity, then affinity registration status cannot be removed and the QUERY TRAN NAME(*tranname*) SHOW(ALL) command will always show the AFFIN status. The following UPDATE TRAN NAME(*tranname*) STOP(SCHD) command will therefore stop both normal and affinity registration.

Note: Because there is no OPTION(AFFIN) keyword for STOP(SCHD), subsequent UPDATE TRAN START(SCHD) commands, with or without the OPTION(AFFIN) keyword, will perform both normal and affinity registration.

The AFFIN status is lost across IMS cold starts. If the DFSMSCE0 user message routing exit is used to set local affinity for an input transaction message in a shared-queues (SQ) environment and the IMS control region is stopped, and subsequently cold started, you will need to issue the

UPDATE TRAN NAME(*tranname*) START(SCHD) OPTION(AFFIN) command for the transaction to be registered with affinity status so the messages will be scheduled.

At the end of cold start processing, when shared-queues informs are completed, the inform for the transaction is done without affinity to the local IMSID because local affinity is set by the exit. Because the inform for the transaction is not part of the transaction definition, it is not maintained across a cold start. As a result, the message with local affinity cannot be scheduled. The UPDATE TRAN START(SCHD) OPTION(AFFIN) command corrects this situation by issuing an inform for the transaction with affinity to the local IMSID.

ALLRSP
Indicates that the response lines are to be returned for all resources that are processed on the command. The default action is to return response lines only for the resources that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SCOPE()
Specifies where IMS should apply the change. The default is ALL.

ALL
Changes are applied to the active IMS systems to which the command is routed. If you specify that global area status is to be maintained, changes are also applied globally by updating the value maintained by RM. The RM status is updated only by the command master IMS. If global area status is not maintained, the command action is same as the SCOPE(ACTIVE) command.

ACTIVE
Changes are applied to the active IMS systems to which the command is routed.

RM maintains global status information for some transactions. IMS updates that information in RM based on commands. When SCOPE(ALL) is specified, every IMS system that processes the command updates information in its local control blocks. Only the IMS command master calls RM to update the information globally.

SET()
Specifies the attribute values to be changed. A transaction must have the same characteristics in all systems where it is defined when it is shared. These characteristics include:

- Nonconversational or conversational
- SPA size if conversational
- Single-segment or multisegment messages
- Non-inquiry or inquiry
- Recoverable or nonrecoverable

AOCMD
Specifies the AOI option that you want to change, which indicates whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call. If AOCMD is defined as CMD, TRAN, or Y, and the AOI1 execute parameter is defined as AOI1=N, no authorization checking is done and the transaction is permitted to issue CMD and ICMD calls.

CMD
Indicates that the transaction is permitted to issue type-1 AOI CMD

calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, authorization checking is based on which transactions can issue a particular command. In this case, the commands (or the first three characters of the commands) need to be defined to RACF or an equivalent product as a user. The type-1 AOI transactions must be defined as profiles under the TIMS class, and for each transaction, the commands it can issue must be specified. Defining AOCMD(CMD) requires you to create fewer user IDs than you need to create for the AOCMD(TRAN) definition. However, defining AOCMD(CMD) requires you to create or modify a larger number of resource profiles.

N Indicates that the transaction is not permitted to issue type-1 AOI CMD calls. The transaction is permitted to issue type-2 AOI ICMD calls.

TRAN
Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the transaction code is used for authorization. The first authorization check results in the accessor environment element (ACEE) being built. This environment is kept for use by future authorization checks. The type-1 AOI transaction needs to be defined to RACF or an equivalent product as a user. The transactions will then be specified on RACF PERMIT statements for each command they are allowed to issue from a type-1 AOI transaction. Specifying AOI transactions as users to RACF might conflict with the name of a user already defined to RACF. If this occurs, then either the transaction name or the existing user name needs to be changed.

Y Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the user ID or the program name is used for authorization. For some environments, if a Get Unique call has not yet occurred, the program name is used for authorization.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

CLASS

Specifies the transaction class, which is an attribute used to select a transaction for scheduling. A transaction can be scheduled if there is a message processing region available for that class. The value can be a number from 1 to 999. This value must not exceed the value given (by specification or default) on the MAXCLAS= keyword of the IMSCTRL macro.

Define CPI-C transactions with a different message class from that used for non-CPI-C transactions. IMS handles all CPI-C transactions as priority zero within the transaction class.

CMTMODE

Specifies when database updates and non-express output messages are committed. This operand affects emergency restart.

MULT
Database updates and non-express output messages are committed only when the application program terminates normally, when the processing limit count has been reached, or, in the case of a pseudo-WFI dependent region, when there are no more messages on the queue. For example, if five transactions are processed during a single schedule of a program, all five are committed only when the fifth one is completed and the program terminates. Until a transaction has been committed, locks for updated database records are not released and non-express output messages are not queued for output. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing.

If the transaction results in the application calling an external subsystem, such as DB2, the Commit Verify exit provided by the external subsystem can determine whether CMTMODE(MULT) is supported. See documentation under the Commit Verify exit routine in *IMS Version 11 Exit Routines*.

SNGL
Database updates and non-express output messages are committed when the application program completes processing each transaction. IMS invokes commit processing either when the application program requests the next message (issues a GU to the IO-PCB), or when the application program terminates. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing. If an application ends abnormally before committing its message, emergency restart requeues the message that was in process before the abend and makes it available for reprocessing.

Keyword combination rules include the following:

- CONV(Y) and CMTMODE(MULT) are mutually exclusive.
- WFI(Y) and CMTMODE(MULT) are mutually exclusive.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

CONV
Specifies the conversation option.

N The transaction is not conversational.

Y The transaction is conversational. The transaction message is destined for a conversational program. A conversational program processes transactions made up of several steps. A conversational program receives a message from a terminal, replies to the terminal, but saves the data from the transaction in a scratchpad area (SPA). When the person at the terminal enters more data, the program has the data it saved from the last message in the SPA, so it can continue processing the request without the person at the terminal having to enter the data again.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- CONV(Y) requires SPASZ and SPATRUNC.
- CMTMODE(MULT) and CONV(Y) are mutually exclusive.
- FP(E) and CONV(Y) are mutually exclusive.
- RECOVER(N) and CONV(Y) are mutually exclusive.
- SPASZ and CONV(N) are mutually exclusive.
- SPATRUNC and CONV(N) are mutually exclusive.

CPRI

Specifies a new value for the current priority of a transaction. The CPRI keyword is not allowed for BMP transactions, because BMP transactions should always have a priority of 0. The new CPRI value takes effect the next time the transaction is scheduled. Valid CPRI parameters are numeric values from 0 to 14.

DCLWA

Specifies the log write-ahead option.

N IMS should not perform log write-ahead. Specify N if input message integrity and the consistency of output messages with associated database updates is not required. DCLWA does not apply to response mode or Fast Path input processing, and is ignored during IMS execution.

Y IMS should perform log write-ahead for recoverable, nonresponse input messages and transaction output messages. This ensures the following:

- A nonresponse input transaction is made recoverable across IMS failures, prior to IMS acknowledging receipt of the input.
- Database changes are made recoverable prior to IMS sending associated output reply messages.
- Information in the log buffers is written to the IMS log, before the associated input acknowledgment or output reply is sent to the terminal.

Define DCLWA(Y) for all VTAM terminal types.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

DIRROUTE

Specifies the MSC directed routing option.

N The application program processing a transaction is not informed of the system which originated the transaction. The name of the originating LTERM is placed in the I/O PCB.

Y The application program processing a transaction is informed of the system which originated the transaction, if MSC directed routing is used in a multiple IMS system configuration. An MSNAME corresponding to a logical path back to the originating system is placed in the I/O PCB.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

EDITRTN

Specifies the 1- to 8-character name of your transaction input edit routine that edits messages before the program receives the message. This name must begin with an alphabetic character. The specified edit routine (load module) must reside on the USERLIB data set before IMS system definition stage 2 execution. This routine cannot be the same one defined by the system definition TYPE EDIT= parameter.

EDITRTN is used for a Fast Path potential transaction when the transaction is routed to IMS.

For input from LU 6.2 devices, the user edit exit routine DFSLUEE0 is called instead of the transaction input edit routine specified in EDIT.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

| To remove an edit routine name from a transaction definition, specify
| EDITRTN(0).

FP(E) and EDITRTN are mutually exclusive.

EDITUC

Specifies the edit to uppercase option.

N The input data is not translated to uppercase. It can consist of uppercase and lowercase characters as entered from the terminal.

Y

The input data is to be translated to uppercase before it is presented to the processing program. If FP(E) or FP(P), the transaction is to be translated to uppercase before being presented to the edit/routing exit routine.

Specifying EDITUC(Y) for VTAM terminals prevents the transmission of embedded device control characters.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

EMHBSZ

Specifies the EMH buffer size required to run the Fast Path transaction. This overrides the EMHL execution parameter. If EMHBSZ is not specified, the EMHL execution parameter value is used. The value can be a number from 0 to 30 720.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- EMHBSZ > 0 requires Fast Path to be defined.
- FP(N) and EMHBSZ > 0 are mutually exclusive.

EXPRTIME

Specifies the elapsed time in seconds that IMS can use to cancel the input transaction. After a transaction is submitted to IMS, the transaction could be delayed for processing because of a stopped transaction or a potential system slow down. In that case, the user or client application could time out before the transaction is processed. When IMS eventually schedules and processes the transaction, the response message is no longer wanted. With the elapsed time specified for the transaction, IMS can flag the input transaction as expired so that the system does not waste CPU cycles to process the unwanted transaction.

The value can be a number, in seconds, which can range from 0 to 65535. The default is 0, which means that no expiration time is set for this transaction. The transaction expiration attribute is supported by all of the IMS TM interfaces.

Restriction: The transaction expiration checking is not performed at the GU time for Fast Path transactions, IMS conversational transactions, and program-to-program switch transactions.

FP Specifies the Fast Path option.

E The transaction is processed exclusively as Fast Path. The program must be defined as Fast Path exclusive.

N The transaction is not a candidate for Fast Path processing. The program must be defined as not Fast Path. When FP(N) is specified, any attempt to use Fast Path resources or commands might yield unpredictable results.

P

The transaction is a potential candidate for Fast Path processing. Fast Path-potential transactions must be able to run under two applications: a Fast Path exclusive application and a non-Fast Path application. A Fast Path exclusive application should be defined to which this transaction can be routed. Fast Path–potential transactions must be processed by a user edit/routing exit to determine whether the transaction is actually to be processed by IMS Fast Path. If it is to be processed by IMS Fast Path, the edit/routing exit routine associates the transaction with a routing code. This routing code identifies which Fast Path application program is to process the transaction.

The program defined by PGM() must not be defined as Fast Path exclusive.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

In order to update a transaction from FP(E) to FP(N) or from FP(N) to FP(E) you must also update the transaction to point to a program that has the same FP() attribute. If you do not update the program attribute, the command will fail because of a program conflict.

If a Fast Path exclusive FP(E) transaction is being updated to Fast Path potential FP(P) or non-Fast Path FP(N) transaction and no limit count is specified, a limit count of 65535 is set.

Keyword combination rules include the following:

- CONV(Y) and FP(E) are mutually exclusive.
- EDITRTN and FP(E) are mutually exclusive.
- EMHBSZ>0 and FP(N) are mutually exclusive.
- FP(E) and FP(P) require Fast Path to be defined.
- MSGTYPE(MULTSEG) and FP(E) are mutually exclusive.
- MSGTYPE(MULTSEG) and FP(P) are mutually exclusive.
- MSNAME and FP(E) are mutually exclusive.
- RECOVER(N) and FP(E) are mutually exclusive.
- RECOVER(N) and FP(P) are mutually exclusive.
- RESP(N) and FP(E) are mutually exclusive.
- RESP(N) and FP(P) are mutually exclusive.
- SIDL and FP(E) are mutually exclusive.
- SIDR and FP(E) are mutually exclusive.

INQ

Specifies the inquiry option.

N This is not an inquiry transaction.

Y

This is an inquiry transaction. If INQ(Y) is specified, you can also specify whether this transaction should be recovered during an IMS emergency or normal restart using the RECOVER() parameter.

This should be specified only for those transactions that, when entered, do not cause a change in any database. Programs are prohibited from issuing ISRT, DLET, or REPL calls to a database when scheduled to process a transaction defined as INQ(Y).

An application program cannot do an SQL INSERT, DELETE, or UPDATE when the IMS transaction is defined with INQ(Y).

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There

must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- RECOVER(N) and INQ(N) are mutually exclusive.

LCT

Specifies the limit count. This is the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction. The value can be a number from 1 to 65535. The default is 65535.

The limit count value is ignored for a transaction processed by a BMP.

The limit count value is ignored in a shared-queues environment.

The limit count value does not apply to FP exclusive transactions and is ignored.

LOCK

Specifies that the LOCK status is to be set on or off. SET(LOCK(ON | OFF)) cannot be specified with any other SET attribute. SET(LOCK(ON | OFF)) can be specified with the START or STOP keyword.

ON Locks the transaction and prevents it from being scheduled. LOCK(ON) cannot be specified for Fast Path exclusive transactions, but can be specified for Fast Path potential transactions. LOCK(ON) cannot be specified for transactions for CPI Communications driven programs.

OFF

Unlocks the transaction and allows it to be scheduled.

LPRI

Specifies the limit priority. This is the scheduling priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The value can be a number from 0 through 14.

When the limit priority is used and the scheduling priority is raised to the limit priority, the priority is not reduced to the normal priority until all messages enqueued for this transaction name are processed. If you do not want the limit priority for this transaction, define equal values for the normal priority and limit priority, and a limit count of 65535.

When a transaction is processed exclusively by a batch message program (BMP), define the limit priority as 0. If the program specified by PGM() is defined with a program type of batch, the current priority is forced to be 0. However, a batch message processing region (BMP) can process transactions with current scheduling priorities other than 0.

This priority also controls the priority of messages created by this transaction and sent to a destination in a remote system. See also the discussion on MSC priorities under the NPRI definition.

The limit priority value is ignored for a transaction processed by a BMP.

The limit priority value is ignored in a shared-queues environment.

MAXRGN

Specifies a new value for the maximum number of regions that can be simultaneously scheduled for a given transaction. The transaction must be eligible for parallel scheduling (load balancing). The value of the MAXRGN parameter must be between 0 and the number specified on the MAXPST=control region parameter.

The keyword limits the number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction. When the number of MPP regions is not limited, one transaction might monopolize all available regions. The value can be a number from 0 to 999. MAXRGN(0) means that no limit is imposed. If you define the application program with a scheduling type of SERIAL, omit the MAXRGN parameter or define it as 0.

The following keyword combinations are mutually exclusive:

- PARLIM(65535) and MAXRGN value greater than 0
- SERIAL(Y) and MAXRGN value greater than 0

MSGTYPE

Specifies the message type (single segment or multiple segment). It specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing.

If MSC-directed routing is used in a multiple IMS system configuration, IMS does not ensure that both the message and the transaction destined to process that message are either single segment or multiple segments.

MULTSEG

The incoming message can be more than one segment in length. It is not eligible for scheduling to an application program until an end-of-message indication is received, or a complete message is created by MFS.

SNGLSEG

The incoming message is one segment in length. It becomes eligible for scheduling when the terminal operator indicates end-of-segment.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- FP(E) and MSGTYPE(MULTSEG) are mutually exclusive.
- FP(P) and MSGTYPE(MULTSEG) are mutually exclusive.

MSNAME

Specifies the one- to eight-character name of the logical link path in a multiple IMS system configuration (MSC). A logical link path is a path between any two IMS systems. The IMS systems are identified by the remote system ID and the local system ID associated with the logical link path. The remote system ID identifies the system in which messages using this path are to be processed. The local system ID identifies this system.

For an UPDATE TRAN command that is changing a transaction to a remote transaction, or changing the MSC path, the new MSNAME must already be defined.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- FP(E) and MSNAME are mutually exclusive.
- SIDL and MSNAME are mutually exclusive.
- SIDR and MSNAME are mutually exclusive.

NPRI

Specifies the normal scheduling priority. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The normal priority is assigned to the transaction as the scheduling priority when the number of input transactions enqueued and waiting to be processed is less than the limit count value. The value can be a number from 0 through 14. The default is 1.

This priority also controls the priority of messages created by this transaction and sent to a destination in a remote system.

When a transaction is processed exclusively by a batch message program (BMP), code the normal priority as 0.

When a transaction is processed exclusively by a batch message program (BMP), define the limit priority as 0. If the application program specified by PGM() is defined with a program type of batch, the current priority is forced to be 0. However, a batch message processing region (BMP) can process transactions with current scheduling priorities other than 0.

For remote transactions, the priority used to send the transaction to the processing system, which is termed *the MSC link message priority*. The three MSC link message priority groups are:

- Low
- Medium
- High

The low priority group consists of primary requests in the input terminal system. This group is assigned remote transaction priorities from 0 to 6. The medium group consists of secondary requests, responses, primary requests in an intermediate system, and primary requests in the input terminal system. This group is assigned a remote transaction priority of 7. The high group consists of primary requests in the input terminal system. Messages in this group are assigned remote transaction priorities from 8 to 14. Within each group, messages have a priority based on the current priority value of the transaction or remote transaction in the input terminal system for primary requests, and on the latest processing system for secondary requests and responses.

In an MSC configuration, the transaction priority determines the priority used to send messages inserted by this transaction across an MSC link. If

the transaction inserts multiple messages to the same destination (for example, pages to a printer) and these messages must be sent in the order inserted, the normal and limit priority values should be the same. If the normal and limit priority values are not identical, messages inserted at a higher priority than previously inserted messages could arrive at their destination first. (This restriction does not apply to multiple segments of the same message.)

The normal priority value is ignored for a transaction processed by a BMP.

PARLIM

Specifies the parallel processing limit count. This is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. This is the threshold value to be used when the associated application is defined with a scheduling type of parallel. An additional region is scheduled whenever the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.

The value can be a number from 0 to 32767 or 65535. PARLIM(0) indicates that any input message can cause a new region to be scheduled because the scheduling condition is always met (the number of messages is greater than zero). If you specify PARLIM(0), you should specify a MAXRGN value to limit the number of regions that can be scheduled to process a particular transaction. PARLIM(65535) means that parallel processing is disabled and IMS allows the transaction to be scheduled in only one region at a time.

The value specified for PARLIM applies to message processing programs (MPPs) only; it is not supported for batch message processing programs (BMPs).

If you define the application as SERIAL or the scheduling type as SERIAL, define PARLIM(65535).

In a shared-queues environment (when the scheduling type is PARALLEL), any PARLIM value other than 65535 causes a new region to be scheduled whenever the successful consecutive GU count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction. For shared-queues environments, the successful consecutive GU count is used instead of the queue count. New regions continue to be scheduled up to the maximum number of regions specified on MAXRGN.

Note: In a shared-queues environment, the PARLIM value behaves differently than it does in a non-shared-queues environment. In a non-shared-queues environment, the queue depth (the number of messages that are currently enqueued) for the transaction is used as the value that is compared with the PARLIM value to determine when to schedule another region. IMS responds to a growing queue of input transactions by scheduling more regions as the queue grows.

In a shared-queues environment, each individual IMS does not know the depth of the queue, because the queue is in the shared-queues coupling facility structure that is managed by Common Queue Server (CQS). The transaction queue might be added to by many different IMS systems. IMS is notified only when the first message is put in a queue (that is, when the queue becomes *not empty*). IMS is not notified for every subsequent

message that is placed on the queue after that first message. In a shared-queues environment, the PARLIM comparison is done against a counter that each IMS keeps of the number of successful consecutive GU calls for the transaction by that IMS, rather than queue depth. IMS schedules more regions when it consistently gets messages from CQS when it asks for them. Thus, in a shared-queues environment, IMS infers the depth of the queue of messages based on processing activity, but it does not know the actual depth of the queue.

A PARLIM value of 0 in a shared-queues environment is the most responsive setting. PARLIM(0) ensures that message regions are scheduled until all messages are processed from the transaction queue, or until the maximum region value (MAXRGN) limit is reached. PARLIM(0) might, however, result in many unnecessary schedules (or *false schedules*). A false schedule occurs when a message region is scheduled and finds no more messages on the queue. This is a consequence of IMS not knowing how many messages are queued on the transaction queue, and having to try to read the queue to see if there are more messages.

Setting the PARLIM to a value greater than 0 can reduce the number of false schedules, because IMS then schedules a new message region only after a number of messages have been obtained consecutively without the queue becoming empty. Setting the PARLIM to a value of 2 or greater is useful for reducing false schedules for transactions that are low-volume and that run relatively quickly (such that the queue depth is typically 1), because it avoids scheduling a second region until the first region has obtained at least two messages in a row. However, be aware that while a PARLIM value greater than 0 can reduce unnecessary schedules, it is also less responsive. If a transaction is long running, or if its processing is delayed (for example, because of locking contention), the consecutive GU count does not change while the transaction is executing, and no additional message regions are scheduled. This can result in delayed processing of other messages for this same transaction until a currently-scheduled message completes. This delay can occur even if message regions are available to process the transaction.

Keyword combination rules include the following:

- MAXRGN>0 and PARLIM(65535) are mutually exclusive.
- SERIAL(Y) and PARLIM between 0 and 32767 are mutually exclusive.

PGM

Specifies the name of the application program associated with the transaction. The program must exist unless the transaction is defined as REMOTE(Y).

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

PLCT

Specifies the processing limit count. This is maximum number of messages sent to the application program by the IMS for processing without reloading the application program. The value must be a number from 0

through 65535. PLCT(0) means the maximum number of messages sent to the application is one and the application program is reloaded before receiving a subsequent message. PLCT(65535) means that no limit is to be placed upon the number of messages processed at a single program load. Values 1 through 65535 are eligible for quick reschedule processing.

The value is used to determine how many messages an application program is allowed to process in a single scheduling cycle. When the application program requests, and receives, the number of messages indicated, any subsequent requests result in one of two things.

1. IMS indicates "no more messages exist" if any of the following conditions is true.
   - The region is not an MPP.
   - The currently scheduled mode is not CMTMODE(SNGL).
   - Equal or higher priority transactions are enqueued for the region.

   IMS might, in fact, have other messages enqueued for the application program. It is the responsibility of the application program to terminate when it receives an indicator that no more messages are available. Termination of the application program makes the region it occupied available for rescheduling. This feature makes it possible for IMS to enable scheduling of higher priority transactions that entered the system while the previous transactions were in process. In addition, if any equal-priority transactions are enqueued, they become eligible for scheduling on a first-in, first-out (FIFO) basis.
2. The region goes through quick reschedule and returns the next message to the application of all of the following conditions are true.
   - The region is an MPP.
   - The transaction is CMTMODE(SNGL).
   - No equal or higher transactions are enqueued.
   - Messages are still enqueued for the application.

PLCTTIME

Specifies the processing limit count time. This is the amount of time (in hundredths of seconds) allowable to process a single transaction (or message). The number specifies the maximum CPU time allowed for each message to be processed in the message processing region.

Batch Message Programs (BMPs) are not affected by this setting.

The value can be a number, in hundredths of seconds, that can range from 1 to 6553500. A value of 6553500 means no time limit is placed on the application program.

If Fast Path is used, this keyword specifies, for a given transaction name, the amount of time (in hundredths of seconds) the program is allowed to process a single transaction message. The time represents real time that elapses during transaction processing (not accumulated task time). Real time is used because the input terminal is in response mode and cannot enter another transaction until the response is sent. In this case PLCT() is ignored.

The value controls application program looping. You are not required to optimize the value for program-transaction execution time. However, the time value assigned should not be less than the expected per-transaction execution time. If the scheduled application program exceeds the product of PLCTTIME() and PLCT(), the application program ends abnormally. If

an IMS STIMER value of 2 is specified on the DFSMPR procedure, the region does not abend until completion of the DL/I call.

Important: The application program must not use STIMER timer services. IMS uses STIMER timer services to time the execution of transactions. If an application program issues an MVS STIMER macro, it cancels the STIMER timer services set by IMS. Use the STIMERM macro instead for application program timer requests.

RECOVER
Specifies the recovery option.

N The transaction should not be recovered.

Y The transaction should be recovered during an IMS emergency or normal restart.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- CONV(Y) and RECOVER(N) are mutually exclusive.
- FP(E) and RECOVER(N) are mutually exclusive.
- FP(P) and RECOVER(N) are mutually exclusive.
- INQ(N) and RECOVER(N) are mutually exclusive.

REMOTE
Specifies the remote option.

N The transaction is not remote. The transaction is local and runs on the local system.

Y The transaction is remote. The transaction runs on a remote system.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Local transactions must have the SIDR value set equal to the SIDL value. When updating a transaction from remote to local, the SIDR and SIDL keywords must be specified, along with REMOTE(N), to set the remote SYSID equal to the local SYSID.

Keyword combination rules include the following:

- REMOTE(Y) requires MSNAME or SIDR and SIDL.

RESP
Specifies the response mode option.

N The transaction is not response mode. For terminals specifying or accepting a default of OPTIONS=TRANRESP, input should not stop after this transaction is entered.

Y The transaction is response mode. The terminal from which the

transaction is entered is held and prevents further input until a response is received. For terminals specifying or accepting a default of OPTIONS=TRANRESP, no additional messages are to be allowed after this transaction is entered until this transaction sends a response message back to the terminal. Response mode can be forced or negated by individual terminal definition. RESP(Y) is ignored during online processing for all terminals that do not operate in response mode.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- FP(E) and RESP(N) are mutually exclusive.
- FP(P) and RESP(N) are mutually exclusive.

SEGNO

Specifies the segment number. This is the maximum number of application program output segments that are allowed into the message queues per Get Unique (GU) call from the application program. The value can be a number from 0 through 65535. If SEGNO(0) is defined, the number of segments is not checked by the online system at execution time.

SEGNO cannot be specified for CPI Communications driven transactions.

SEGSZ

Specifies the segment size. This is the maximum number of bytes allowed in any one output segment. The value can be a number from 0 through 65535. If SEGSZ(0) is defined, the segment size is not checked by the online system at execution time.

The maximum output message segment to an LU 6.2 device is 32767. If a transaction is expected to send output to an LU 6.2 device, the SEGSIZE parameter should be no greater than 32767. However, this is not enforced during processing of the command, because IMS cannot determine the device type for the message destination until output time.

SEGSZ cannot be specified for CPI Communications driven transactions.

SERIAL

Specifies the serial option.

N Messages for the transaction are not processed serially. Message processing can be processed in parallel. Messages are placed on the suspend queue after a U3303 pseudoabend. Scheduling continues until repeated failures result in the transaction being stopped with a USTOP.

Y Messages for the transaction are processed serially. U3303 pseudoabends do not cause the message to be placed on the suspend queue but rather on the front of the transaction message queue, and the transaction is stopped with a USTOP. The USTOP of the transaction is removed when the transaction or the class is started with a /START or UPD TRAN command.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a

message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- MAXRGN>0 and SERIAL(Y) are mutually exclusive.
- PARLIM value 0 - 32767 and SERIAL(Y) are mutually exclusive.

SIDL

Specifies the system identification (SYSID) of the local system in a multiple-IMS system (MSC) configuration. The local system is the originating system to which responses are returned. The value can be a number from 1 to 2036, if MSC is enabled, or 0, if MSC is not enabled. The local SYSID can be defined in any or all of the MSNAMEs or transactions.

The SIDL parameter is independent of the link type (CTC, MTM, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- FP(E) and SIDL are mutually exclusive, unless SIDL and SIDR are specified as a pair and are equal to the local system ID of this IMS.
- MSNAME and SIDL are mutually exclusive.
- SIDL value must be defined to this IMS.

SIDR

Specifies the system identification (SYSID) of the remote system in a multiple-IMS system (MSC) configuration. The remote system is the system on which the application program executes. The value can be a number from 1 to 2036, if MSC is enabled, or 0, if MSC is not enabled. The remote SYSID specified must also be defined for an MSNAME.

The SIDR parameter is independent of the link type (CTC, MTM, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement.

Local transactions must have the SIDR value set equal to the SIDL value. When updating a transaction from remote to local, the remote SYSID must be set to the local SYSID.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- FP(E) and SIDR are mutually exclusive, unless SIDL and SIDR are specified as a pair and are equal to the local system ID of this IMS.
- MSNAME and SIDR are mutually exclusive.
- SIDR value must be defined to this IMS.

SPASZ

Specifies the scratchpad area (SPA) size, in bytes, for a conversational transaction. The value can be a number from 16 and 32767.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- CONV(N) and SPASZ are mutually exclusive.
- FP(E) and SPASZ are mutually exclusive.

SPATRUNC

Specifies the scratchpad area (SPA) truncation option of a conversational transaction. This defines whether the SPA data should be truncated or preserved across a program switch to a transaction that is defined with a smaller SPA.

When a conversation initially starts, and when a program is switched, the SPATRUNC option is checked and set or reset as specified. When the option is set, it remains set for the life of the conversation, or until a program switch occurs to a transaction that specifies the option is to be Reset.

When a program switch occurs, the truncated data option for the new transaction is first checked, and that specification is set for the conversation and is used for the SPA inserted into the output message. If the option is not specified for the new transaction, the option currently in effect for the conversation is used.

S IMS preserves all of the data in the SPA, even when a program switch is made to a transaction that is defined with a smaller SPA. The transaction with the smaller SPA does not see the truncated data, but when the transaction switches to a transaction with a larger SPA, the truncated data is used.

R The truncated data is not preserved.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- CONV(N) and SPATRUNC are mutually exclusive.
- FP(E) and SPATRUNC are mutually exclusive.

TRANSTAT

Specifies whether transaction level statistics should be logged for message driven programs. If Y is specified, transaction level statistics are written to the log in a X'56FA' log record.

N Transaction level statistics should not be logged.

Y Transaction level statistics should be logged.

The TRANSTAT keyword on the UPDATE TRAN command gives the user the ability to override the system default or the current value of the TRANSTAT parameter. If the TRANSTAT keyword is omitted on the UPDATE TRAN command, the current transaction level statistics setting is unchanged.

WFI

Specifies the wait-for input option. This attribute does not apply to Fast Path transactions, which always behave as wait-for-input transactions.

N This is not a wait-for-input transaction.

Y This is a wait-for-input transaction. A message processing or batch processing application program that processes WFI transactions are scheduled and invoked normally. If the transaction to be processed is defined as WFI, the program is allowed to remain in main storage after it has processed the available input messages. The QC status code (no more messages) is returned to the program if the PROCLIM count (PLCT) is reached; a command is entered to change the status of the scheduled transaction, database, program, or class; commands relating to the databases used by the transaction are entered, or IMS is terminated with a checkpoint shutdown.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- MODE(MULT) and WFI(Y) are mutually exclusive.

START()

Specifies the attributes to be started.

Q Updates local and global status to start queuing transactions. This change applies to transactions that are entered either from a network or terminal, or using the QUEUE TRAN command, to DB/DC or DCCTL environments.

SCHD

Updates local and global status to start transactions from being scheduled. This change applies to transactions scheduled in DB/DC or DCCTL environments.

In a shared-queues environment, the UPD TRAN START(SCHD) command will result in IMS registering interest for the transaction, which indicates that the transaction can be scheduled at that IMS. An UPDATE TRAN NAME(*) START(SCHD) command does not register transactions that are already registered to CQS.

SUSPEND

If the transaction has messages on the suspend queue, that suspend queue is automatically transferred to the ready queue.

TRACE

Starts the transaction trace, which captures the DL/I portion of Data

Communications (DC) for the specified transaction. The information is written as a 6701 log record to the IMS log.

STOP()
Specifies the attributes to be stopped.

Q Updates local and global status to stop transactions from being queued. This change applies to transactions that are entered either from a network or terminal, or using the QUEUE TRAN command, to DB/DC or DCCTL environments.

SCHD
Updates local and global status to stop transactions from being scheduled. This change applies to transactions scheduled in DB/DC or DCCTL environments.

The UPDATE TRAN STOP(SCHD) command stops the scheduling of transactions; however, the transactions will continue to be processed until the limit count is reached. If the limit count is large, the processing interval will be long.

In a shared-queues environment, the UPD TRAN STOP(SCHD) command results in IMS deregistering interest for the transaction, which indicates that the transaction cannot be scheduled at the IMS.

TRACE
Stops the transaction trace.

Usage notes

Resources exist for the life of the IMS unless they are deleted using a DELETE command. Resources are recoverable across an IMS warm start or emergency restart. Resources are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE TRAN command is recoverable across an IMS warm start or emergency restart, except for START(TRACE) and STOP(TRACE). Updates to the definitional attributes of a transaction are lost if IMS is cold started, unless the transaction definitions are exported to an RDDS and then imported from the RDDS at the cold start.

The UPDATE TRAN command can be issued only through the OM API. This command applies to DB/DC and DCCTL systems. This command is not valid on the XRF alternate, RSR tracker, or FDBR region.

The UPDATE TRAN command changes a MODBLKS transaction to dynamic, if the AOCMD, CMTMODE, CONV, DCLWA, DIRROUTE, EDITRTN, EDITUC, EMHBSZ, FP, INQ, MSGTYPE, MSNAME, PLCTTIME, PGM, RECOVER, REMOTE, RESP, SERIAL, SPASZ, SPATRUNC, TRANSTAT, or WFI value is changed.

Each transaction is updated individually, unlike the online change process where either all transactions are updated or no transactions are updated. Some runtime resource definition values for a transaction can be updated only if the transaction is not in use.

You can update the status of a transaction (LOCK, START, STOP) while the transaction is in use.

The following transaction attributes cannot be updated if online change for MODBLKS is enabled: AOCMD, CMTMODE, CONV, DCLWA, DEFAULT, DIRROUTE, EDITRTN, EDITUC, EMHBSZ, FP, INQ, MSGTYPE, MSNAME, PGM, PLCTTIME, RECOVER, REMOTE, RESP, SERIAL, SIDL, SIDR, SPASZ, SPATRUNC, WFI.

If all the attributes specified by the UPDATE command are already defined for the resource, no update is actually made, no resources are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

In a shared message queues environment, the UPDATE TRAN command updates the transaction even if there are messages on the shared messages queues for the transaction. If the messages were placed on the shared message queues using conversation, Fast Path, response mode, or serial transaction attributes that differ from the transaction being updated on this IMS, this IMS will not be able to access the messages on the shared message queues.

The UPDATE TRAN NAME(tranname) SET(REMOTE(N),SIDR(localsysid),SIDL(localsysid)) command sets the remote system ID to be the same as the local system ID. It is equivalent to the command /MSASSIGN TRAN tranname TO LOCAL. The UPDATE TRAN NAME(tranname) SET(REMOTE(N)) command specified without SIDR/SIDL or MSNAME does not change the local and remote system IDs, therefore it is not equivalent to the /MSASSIGN TRAN tranname TO LOCAL command.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE TRAN command and the type-1 IMS commands that perform similar functions.

Table 405. Type-1 equivalents for the UPDATE TRAN command

UPDATE command	Similar IMS type-1 command
UPDATE TRAN(name) START(Q) STOP(SCHD)	/PSTOP TRAN *name*
UPDATE TRAN NAME(name) START(SCHD) STOP(Q)	/PURGE TRAN *name*
UPDATE TRAN NAME(name) START(Q,SCHD,SUSPEND)	/START TRAN *name*
UPDATE TRAN NAME(name) STOP(Q,SCHD)	/STOP TRAN *name*
UPDATE TRAN NAME(name) START(TRACE)	/TRACE SET ON TRAN *name*
UPDATE TRAN NAME(name) STOP(TRACE)	/TRACE SET OFF TRAN *name*
UPDATE TRAN NAME(name) SET(CLASS(*new_class_number*))	/ASSIGN TRAN *name* TO CLS *new_class_number*
UPDATE TRAN NAME(name) SET(CPRI(*new_current_priority*))	/ASSIGN CPRI *new_current_priority* TO TRAN *name*
UPDATE TRAN NAME(name) SET(LCT(*new_limit_count*))	/ASSIGN LCT *new_lmct_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(LPRI(*new_limit_priority*))	/ASSIGN LPRI *new_lpri_number* TO TRAN *name*

Table 405. Type-1 equivalents for the UPDATE TRAN command (continued)

UPDATE command	Similar IMS type-1 command
UPDATE TRAN NAME(name) SET(NPRI(*new_normal_priority*))	/ASSIGN NPRI *new_npri_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(PARLIM(*new_parallel_limit*))	/ASSIGN PARLIM *new_parmlim_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(PLCT(*new_processing_limit*))	/ASSIGN PLCT *new_plmct_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(SEGNO(*new_segment_number*))	/ASSIGN SEGNO *new_segno_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(SEGSZ(*new_segment_size*))	/ASSIGN SEGSZ *new_segsize_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(MAXRG(*new_max_regions*))	/CHANGE TRAN *name* MAXRGN *new_maxrgn_number*

Output fields

The following table shows the UPDATE TRAN output fields. The columns in the table are as follows:

`Short label`
Contains the short label generated in the XML output.

`Keyword`
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

`Meaning`
Provides a brief description of the output field.

Table 406. Output fields for the UPDATE TRAN command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
CONVID	TRAN	Conversation ID of conversation associated with transaction that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
GBL	SCOPE(ALL)	Indicates that the response line is for the global update.
LU	TRAN	APPC LU name associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.

Table 406. Output fields for the UPDATE TRAN command (continued)

Short label	Keyword	Meaning
MBR	N/A	The IMSplex member that built the output line. The IMS identifier of the IMS for which the transaction information is displayed. The IMS identifier is always returned.
NODE	TRAN	Node name of static node associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.
TMEM	TRAN	OTMA tmember name associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.
TPIP	TRAN	OTMA tpipe name associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.
TRAN	N/A	The transaction name. The transaction name is always displayed.
USER	TRAN	User name of dynamic user associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 407. Return and reason codes for the UPDATE TRAN command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See Table 408 on page 823 for details.
X'00000004'	X'00002008'	Invalid number of keywords. Either a SET, START, or STOP keyword is required.
X'00000008'	X'0000200C'‘	No resources were found to be updated. The resource names specified might be invalid or there were no resources that match the filter specified. Confirm that the UPDATE TRAN command is issued with valid resources.
X'00000008'	X'00002040'	More than one filter value is specified on the UPDATE TRAN command. Confirm that only one of SET or START | STOP filters are specified on the command.

Table 407. Return and reason codes for the UPDATE TRAN command (continued)

Return code	Reason code	Meaning
X'00000008'	X'00002044'	The UPDATE TRAN command is not processed because the same attribute value was specified for the START and STOP filters. The attribute "Q,SCHD" can be specified only on START or STOP but not both. For example, UPDATE TRAN START(Q) STOP(Q) is not valid but UPDATE TRAN START(Q) STOP(SCHD) is valid. Confirm that only one START | STOP attribute is specified on the command.
X'00000008'	X'00002048'	The UPDATE TRAN command is not processed because an invalid SET attribute is specified. Confirm that the correct SET attribute is specified on the command.
X'00000008'	X'0000204C'	The UPDATE TRAN command is not processed because a CLASS value specified is invalid. Confirm that the correct CLASS value is specified on the command.
X'00000008'	X'00002050'	The UPDATE TRAN command is not processed because the CPRI value specified is invalid. Confirm that the correct CPRI value is specified on the command.
X'00000008'	X'00002054'	The UPDATE TRAN command is not processed because the LCT (limit count) value specified is invalid. Confirm that the correct LCT value is specified on the command.
X'00000008'	X'00002058'	The UPDATE TRAN command is not processed because the LPRI value specified is invalid. Confirm that the correct LPRI value is specified on the command.
X'00000008'	X'0000205C'	The UPDATE TRAN command is not processed because the MAXGN value specified is invalid. Confirm the correct MAXRGN value is specified on the command.
X'00000008'	X'00002060'	The UPDATE TRAN command is not processed because the NPRI value specified is invalid. Confirm that the correct NPRI value is specified on the command.
X'00000008'	X'00002064'	The UPDATE TRAN command is not processed because the PARLIM value is invalid. Confirm that the PARLIM value is specified on the command.
X'00000008'	X'00002068'	The UPDATE TRAN command is not processed because the PLCT value is invalid. Confirm that the correct PLCT value is specified on the command.
X'00000008'	X'00002100'	CMTMODE(MULT) mutually exclusive with WFI(Y).
X'00000008'	X'00002101'	CONV(Y) mutually exclusive with CMTMODE(MULT).
X'00000008'	X'00002103'	CONV(N) mutually exclusive with SPASZ>0 and SPATRUNC.
X'00000008'	X'00002104'	CONV(Y) mutually exclusive with RECOVER(N).
X'00000008'	X'00002105'	CONV(Y) requires SPASZ and SPATRUNC.
X'00000008'	X'00002108'	Invalid EDITRTN name.

Table 407. Return and reason codes for the UPDATE TRAN command (continued)

Return code	Reason code	Meaning
X'00000008'	X'0000210A'	Invalid EMHBSZ. One of the following situations occurred: • The EMHB size specified is either greater than the maximum size, which is 30720 • The EMHB size specified plus the length of the X'5901' log record prefix is greater than the log buffer size
X'00000008'	X'0000210C'	FP(E) mutually exclusive with EDITRTN.
X'00000008'	X'0000210E'	FP(E) or FP(P) mutually exclusive with MSC keyword MSNAME or SIDR and SIDL.
X'00000008'	X'0000210F'	FP(E) or FP(P) mutually exclusive with MSGTYPE(MULTSEG).
X'00000008'	X'00002110'	FP(N) mutually exclusive with EMHBSZ > 0.
X'00000008'	X'00002111'	FP(E) or FP(P) mutually exclusive with RECOVER(N).
X'00000008'	X'00002112'	FP(E) or FP(P) mutually exclusive with RESP(N).
X'00000008'	X'00002116'	INQ(N) mutually exclusive with RECOVER(N).
X'00000008'	X'00002119'	MSC keyword MSNAME or SIDL/SIDR mutually exclusive with application program defined as Fast Path exclusive (FP(E)) associated with this transaction.
X'00000008'	X'0000211A'	Invalid MSNAME name.
X'00000008'	X'0000211B'	MSNAME mutually exclusive with SIDL and SIDR.
X'00000008'	X'0000211D'	MAXRGN>0 mutually exclusive with PARLIM(65535).
X'00000008'	X'0000211E'	MAXRGN>0 mutually exclusive with SERIAL(Y).
X'00000008'	X'00002121'	PARLIM value mutually exclusive with SERIAL(Y).
X'00000008'	X'00002125'	If REMOTE(Y) is specified, the SIDR value must be a remote SYSID and the SIDL value must be a local SYSID. If neither the MSNAME keyword nor the SIDR and SIDL keywords are specified explicitly, the SIDR and SIDL values from the existing transaction definition are used. If REMOTE(N) is specified, the SIDR value must be equal to the SIDL value. If the SIDR and SIDL values are not specified explicitly, the values from the existing transaction definition are used. The MSNAME keyword cannot be specified with REMOTE(N).
X'00000008'	X'00002126'	Invalid SIDL value.
X'00000008'	X'00002127'	SIDL/SIDR must be specified as a pair. Either SIDL was specified alone or SIDR was specified alone.
X'00000008'	X'00002128'	Invalid SIDR value.
X'0000000C'	X'00003000'	Command was successful for some of the resources. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), output lines are only returned for resources with nonzero completion codes. See Table 408 on page 823 for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See Table 408 on page 823 for details.

Table 407. Return and reason codes for the UPDATE TRAN command (continued)

Return code	Reason code	Meaning
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined, FP(E), FP(P), or EMHBSZ >0 invalid.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004310'	Storage could not be obtained for the Transaction Input edit routine table. A cold start is required to fix this error.
X'00000010'	X'00004314'	The Transaction Input edit routine could not be loaded.
X'00000010'	X'00004318'	A new Transaction Input edit routine could not be added. The maximum of 255 routines has already been reached.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.
X'00000014'	X'00005010'	Latch could not be obtained.
X'00000014'	X'000050FF'	The UPDATE TRAN or UPDATE TRANDESC command processing terminated because of an internal error.

Errors unique to the processing of the UPDATE TRAN command are returned as a completion code. A completion code is returned for each action against an individual resource. The completion codes in the following table might be returned on the UPDATE TRAN command.

Table 408. Completion codes for the UPDATE TRAN command

Completion code	Completion code text	Meaning
0		Command completed successfully for transaction.
10	NO RESOURCES FOUND	Transaction name is invalid, or the wildcard parameter specified does not match any resource names.
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this transaction. This could also mean this command, if the resource is specified by more than one specific or wildcard parameter. Or, the transaction is updating the program name and another command is in progress for that program.
19	CMTMODE(MULT)/ WFI(Y) CONFLICT	Transaction update failed because commit mode multiple CMTMODE(MULT) option conflicts with the wait-for-input WFI(Y) option.

Table 408. Completion codes for the UPDATE TRAN command (continued)

Completion code	Completion code text	Meaning
1A	IN CONVERSATION	Transaction is in conversation. The conversation ID and terminal in conversation are returned separately. The terminal can be a static node, node and user, dynamic user, APPC luname, or OTMA tmember and tpipe. Suggested actions: Terminate the conversation.
1B	CONV(Y)/ CMTMODE(MULT) CONFLICT	Transaction update failed because the conversation CONV(Y) option conflicts with the commit mode multiple CMTMODE(MULT) option.
1E	CONV(N)/SPASZ OR SPATRUNC CONFLICT	Transaction update failed because the non-coversation CONV(N) option conflicts with the SPA size or SPA truncation option.
1F	CONV(Y)/ RECOVER(N) CONFLICT	Transaction update failed because the conversation CONV(Y) option conflicts with the nonrecoverable RECOVER(N) option.
2F	FP(E)/ BMPTYPE(Y) CONFLICT	Transaction update failed because Fast Path exclusive FP(E) option conflicts with program defined as batch BMPTYPE(Y).
30	QUEUE-ONLY TRANSACTION	The UPDATE TRAN command is invalid for the resource because the transaction is a queue-only transaction.
34	NOT ALLOWED FOR A CPIC TRAN	The UPDATE TRAN command is invalid for the resource because the transaction is a CPIC transaction.
35	FP(E)/EDITRTN CONFLICT	Transaction update failed because Fast Path exclusive FP(E) option conflicts with the edit routine EDITRTN.
36	FP(E)/FP(N) PGM CONFLICT	Transaction update failed because Fast Path exclusive FP(E) option conflicts with program defined as non-Fast Path FP(N).
38	NOT ALLOWED FOR A BMP	The UPDATE TRAN command is invalid for the resource because the PSB associated with the transaction is a BMP.
3A	FP(E OR P)/MSC KEYWORD CONFLICT	Transaction update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with the MSC MSNAME, SIDR/SIDL option.
3B	FP(E OR P)/ MSGTYPE(MULTSEG) CONFLICT	Transaction update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with message type multiple segment MSGTYPE(MULTSEG) option.
3C	MAXRGN>0/ PARLIM(65535) CONFLICT	Transaction update failed because maximum region count MAXRGN value conflicts with the parallel limit count PARLIM value 65535, which means parallel scheduling is disabled. MAXRGN > 0 is not allowed with PARLIM(65535).

Table 408. Completion codes for the UPDATE TRAN command (continued)

Completion code	Completion code text	Meaning
3E	FP(N)/FP(E) PGM CONFLICT	Transaction update failed because non-Fast Path FP(N) option conflicts with program defined as Fast Path exclusive FP(E).
3F	FP(P)/BMPTYPE(Y) CONFLICT	Transaction update failed because Fast Path potential program FP(P) conflicts with the program defined as batch BMPTYPE(Y).
40	PARLIM/ SCHDTYPE(SERIAL) CONFLICT	The PARLIM cannot be changed for the resource because the PSB associated with the transaction is defined as does not have parallel scheduling.
41	FP(E OR P)/ RECOVER(N) CONFLICT	Transaction update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with nonrecoverable RECOVER(N) option.
42	FP(E OR P)/RESP(N) CONFLICT	Transaction update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with response mode RESP(N) option.
44	TRANSACTION BUSY	The UPDATE TRAN command cannot be processed for the resource because the transaction is currently being scheduled.
45	INVALID SIDR VALUE	The UPDATE TRAN command could not be completed for the resource because the SID number is invalid.
49	INQ(N)/RECOVER(N) CONFLICT	Transaction update failed because non-inquiry INQ(N) option conflicts with nonrecoverable RECOVER(N) option.
4A	IN USE	Transaction is in use. Queuing is in progress, either terminal input or a program-to-program switch.
4F	INVALID MAXRGN VALUE	Maximum region MAXRGN value is invalid.
5E	MAXRGN>0/ SERIAL(Y) CONFLICT	Transaction update failed because non-zero maximum region value conflicts with serial SERIAL(Y) option.
61	DFSBCB STORAGE ERROR.	DFSBCB storage could not be obtained.
6A	FP(P)/FP(E) PGM CONFLICT	Transaction to be updated as Fast Path potential FP(P) conflicts with program already defined as Fast Path exclusive FP(E).
6B	PARLIM/SERIAL(Y) CONFLICT	Transaction update failed because the parallel limit PARLIM value conflicts with the serial SERIAL(Y) option.
6D	INVALID PROGRAM NAME	Program name specified is invalid.
73	PROGRAM SCHEDULED	Program is scheduled.

Table 408. Completion codes for the UPDATE TRAN command (continued)

Completion code	Completion code text	Meaning
79	REMOTE/SIDR/SIDL/ MSNAME CONFLICT	Transaction update failed because there is a conflict between the REMOTE value and the MSNAME keyword or the SIDR and SIDL values.
85	SUSPENDED	Transaction is on the suspend queue.
87	TRAN QUEUEING	Transaction has messages queued (non-shared-queues environment).
89	TRAN SCHEDULED	Transaction is scheduled.
90		UPD TRAN command did not complete because of an internal error.
99	NOT INITIALIZED	Transaction update failed because the transaction was not successfully initialized. QUERY TRAN STATUS(NOTINIT) displays the reason the transaction is not initialized, for example the program does not exist. Correct this problem and issue an UPDATE TRAN START(SCHD,Q) command to initialize the transaction.
9B	FASTPATH TRAN NOT SUPPORTED	Transaction update failed because the transaction is Fast Path exclusive FP(E).
B3	TRAN ELIGIBLE FOR SCHEDULING	Transaction is eligible for scheduling and cannot be updated. You might need to stop the transaction with the UPDATE TRAN STOP(Q,SCHD) command before attempting the UPDATE again.
B5	ROUTING CODE ALREADY EXISTS	Transaction update failed because Fast Path exclusive FP(E) option conflicts with routing code that already exists by that transaction name.
BF	FP(E)/CONV keyword conflict	Fast Path exclusive FP(E) is mutually exclusive with any conversation keyword, including CONV(Y), SPASZ, and SPATRUNC. An FP exclusive transaction cannot be defined as conversational.
145	MESSAGE IN PROGRESS ACROSS LINK	Transaction update failed because a message for the transaction is in progress across the MSC link.

Examples

The following are examples of the UPDATE TRAN command:

Example 1 for UPDATE TRAN command

TSO SPOC input:

```
UPDATE TRAN NAME(BADNAME,AOBMP,APOL17,CPI%,BAD*) SET(WFI(Y))
```

TSO SPOC output:

```
Response for: UPDATE TRAN NAME(BADNAME,AOBMP,APOL17,CPI%,BAD*)
Trancode MbrName    CC CCText
AOBMP    IMS1        0
APOL17   IMS1       19 CMTMODE(MULT)/WFI(Y) CONFLICT
BAD*     IMS1       10 NO RESOURCES FOUND
BADNAME  IMS1       10 NO RESOURCES FOUND
CPI1     IMS1        0
CPI2     IMS1        0
CPI3     IMS1        0
CPI4     IMS1        0
```

OM API input:

```
CMD(UPDATE TRAN NAME(BADNAME,AOBMP,APOL17,CPI%,BAD*) SET(WFI(Y)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.312 00:07:49.667460</statime>
<stotime>2006.312 00:07:49.668362</stotime>
<staseq>BFAC164BEAE846C0</staseq>
<stoseq>BFAC164BEB20A400</stoseq>
<rqsttkn1>USRT011 10160749</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>TRAN            </kwd>
<input>UPDATE TRAN NAME(BADNAME,AOBMP,APOL17,CPI%,BAD*,AOBMP)
SET(WFI(Y)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="GBL" llbl="Global" scope="GBL" sort="d" key="2" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="y" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="CONVID" llbl="ConvID" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="NODE" llbl="NodeName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="USER" llbl="UserName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
```

```
<hdr slbl="LU" llbl="LUName" scope="LCL" sort="n" key="0"
 scroll="yes" len="24" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMEM" llbl="TMember" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TPIP" llbl="TPipe" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>TRAN(BADNAME ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>TRAN(AOBMP   ) MBR(IMS1) CC(   0) </rsp>
<rsp>TRAN(APOL17  ) MBR(IMS1) CC(  19) CCTXT(CMTMODE(MULT)/WFI(Y)
 CONFLICT) </rsp>
<rsp>TRAN(CPI1    ) MBR(IMS1) CC(   0) </rsp>
<rsp>TRAN(BAD*    ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>TRAN(CPI2    ) MBR(IMS1) CC(   0) </rsp>
<rsp>TRAN(CPI3    ) MBR(IMS1) CC(   0) </rsp>
<rsp>TRAN(CPI4    ) MBR(IMS1) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update several transactions to be wait-for-input. The update succeeded for several transactions, as shown by the completion code 0. The update failed for transaction APOL17 with completion code 19, since the WFI(Y) attribute conflicts with the CMTMODE(MULT) attribute already defined for APOL17. The update fails for transaction BADNAME and for parameter BAD* with completion code 10, since transaction BADNAME does not exist and no transaction name starts with BAD.

Example 2 for UPDATE TRAN command

TSO SPOC input:

```
UPDATE TRAN NAME(APOL12) START(SCHD) OPTION(AFFIN)
```

TSO SPOC output:

```
Trancode MbrName    CC
APOL12   IMS1        0
```

Explanation: Transaction APOL12 is started with affinity.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related tasks:

Updating runtime transaction resource and descriptor definitions with the UPDATE command (System Definition)

Related reference:

/LOCK command (Commands)

Command keywords and their synonyms (Commands)

Chapter 3, “/PSTOP command,” on page 19
“/START TRAN command” on page 498
“/STOP TRAN command” on page 551
“/TRACE TRAN command” on page 614
“/UNLOCK TRAN command” on page 627

Commit Verify exit routine (Exit Routines)

UPDATE TRANDESC command

Use the UPDATE TRANDESC command to update transaction descriptors. A descriptor is a model that can be used to create descriptors or resources.

Updating a descriptor changes only those attributes explicitly specified on the UPDATE command. Attributes not specified retain their existing values. Any transaction resource or descriptor can be created using this descriptor as a model, by specifying the CREATE LIKE(DESC(*descriptor_name*)) command. Any descriptor or resource that was already created using this descriptor is not updated.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 831
- "Usage notes" on page 844
- "Output fields" on page 845
- "Return, reason, and completion codes" on page 845
- "Examples" on page 849

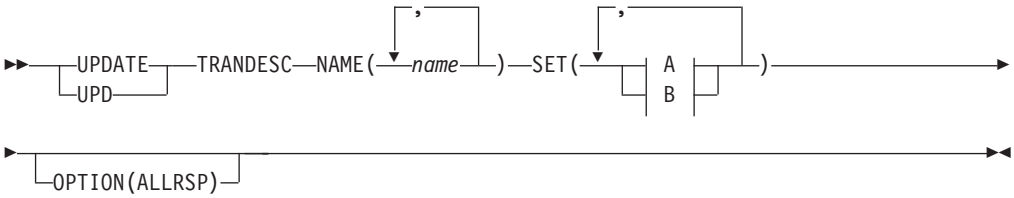
Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which you can use the commands and keywords.

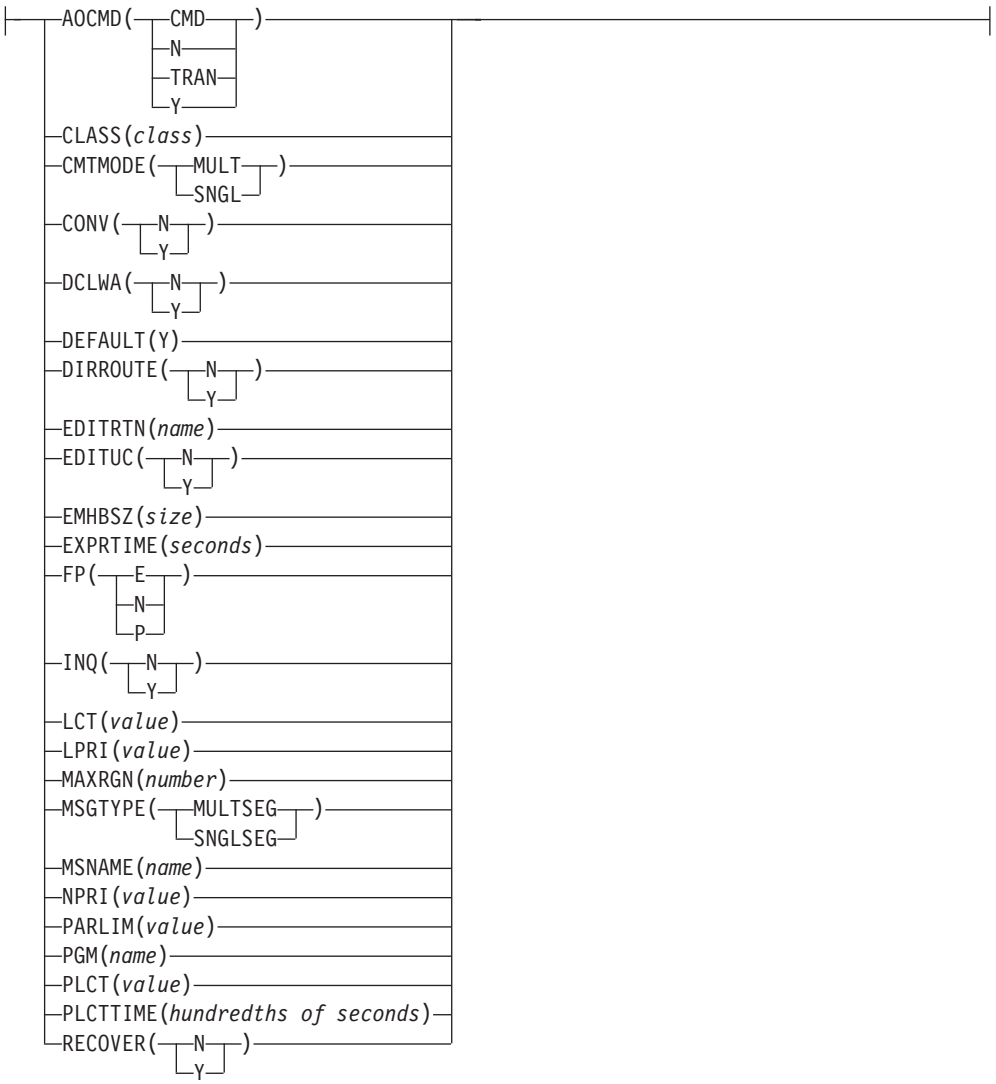
Table 409. Valid environments for the UPDATE TRANDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE TRANDESC	X		X
NAME	X		X
OPTION	X		X
SET	X		X

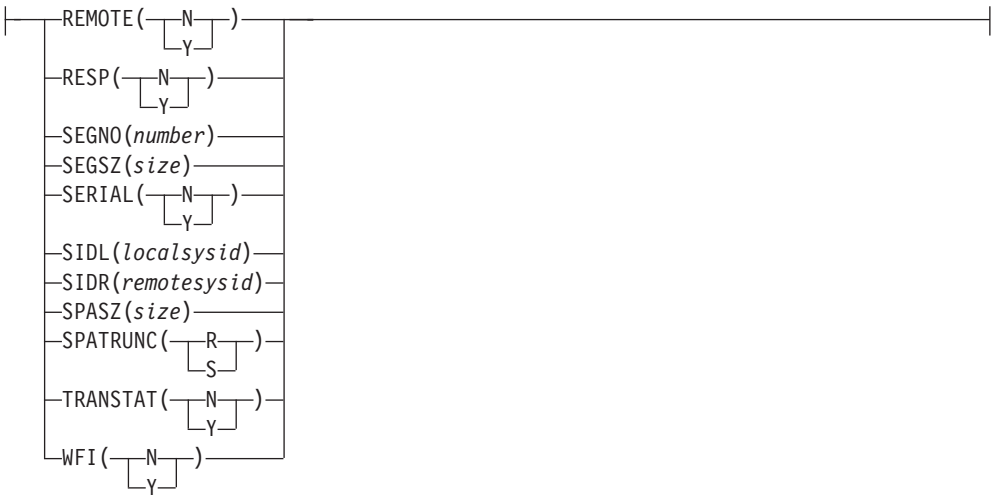
Syntax



A:


AOCMD(CMD|N|TRAN|Y)
CLASS(class)
CMTMODE(MULT|SNGL)
CONV(N|Y)
DCLWA(N|Y)
DEFAULT(Y)
DIRROUTE(N|Y)
EDITRTN(name)
EDITUC(N|Y)
EMHBSZ(size)
EXPRTIME(seconds)
FP(E|N|P)
INQ(N|Y)
LCT(value)
LPRI(value)
MAXRGN(number)
MSGTYPE(MULTSEG|SNGLSEG)
MSNAME(name)
NPRI(value)
PARLIM(value)
PGM(name)
PLCT(value)
PLCTTIME(hundredths of seconds)
RECOVER(N|Y)

B:


REMOTE(N|Y)
RESP(N|Y)
SEGNO(number)
SEGSZ(size)
SERIAL(N|Y)
SIDL(localsysid)
SIDR(remotesysid)
SPASZ(size)
SPATRUNC(R|S)
TRANSTAT(N|Y)
WFI(N|Y)

Keywords

The following keywords are valid for the UPDATE TRANDESC command:

NAME
Specifies the 1-8 character name of the transaction descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the descriptor names that are processed. For NAME(*), command responses are returned only for the descriptor names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the descriptor names that are processed.

OPTION
Specifies additional functions to be performed along with the command.

ALLRSP
Indicates that the response lines are to be returned for all descriptors that are processed on the command. The default action is to return response lines only for the descriptors that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SET
Specifies the attribute values to be changed. A transaction must have the same characteristics in all systems where it is defined when it is shared. These characteristics include:

- Nonconversational or conversational
- SPA size if conversational
- Single-segment or multi-segment messages
- Non-inquiry or inquiry
- Recoverable or nonrecoverable

AOCMD
Specifies that the AOI option indicating whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call. If AOCMD is defined as CMD, TRAN, or Y, and the AOI1 execute parameter is defined as AOI1=N, no authorization checking is done, and the transaction is permitted to issue CMD and ICMD calls.

CMD
Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, authorization checking is based on which transactions can issue a particular command. In this case, the commands (or first three characters of the commands) need to be defined to RACF or equivalent product as a user. The type-1 AOI transactions must be defined as profiles under the TIMS class, and for each transaction, the commands it can issue must be specified. Defining AOCMD(CMD) requires you to create fewer user IDs than you need to create for the AOCMD(TRAN) definition. However, defining AOCMD(CMD) requires you to create or modify a larger number of resource profiles.

N Indicates that the transaction is not permitted to issue AOI type-1 CMD calls. The transaction is permitted to issue AOI type-2 ICMD calls.

TRAN
Indicates that the transaction is permitted to issue type-1 AOI CMD

calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the transaction code is used for authorization. The first authorization check results in the accessor environment element (ACEE) being built. This environment is kept for use by future authorization checks. The type-1 AOI transaction needs to be defined to RACF or equivalent product as a user. The transactions will then be specified on RACF PERMIT statements for each command they are allowed to issue from a type-1 AOI transaction. Specifying AOI transactions as users to RACF might conflict with the name of a user already defined to RACF. If this occurs, then either the transaction name or the existing user name needs to be changed.

Y Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the user ID or program name is used for authorization. For some environments, if a Get Unique call has not yet happened, then the program name is used for authorization.

CLASS

Specifies the transaction class, which is an attribute used to select a transaction for scheduling. A transaction can be scheduled if there is a message processing region available for that class. The value can be a number from 1 to 999. This value must not exceed the value given (by specification or default) on the MAXCLAS= keyword of the IMSCTRL macro.

Define CPI-C transactions with a different message class from that used for non-CPI-C transactions. IMS handles all CPI-C transactions as priority zero within the transaction class.

CMTMODE

Specifies when database updates and non-express output messages are committed. This operand affects emergency restart.

MULT
Database updates and non-express output messages are committed only when the application program terminates normally, when the processing limit count has been reached, or, in the case of a pseudo-WFI dependent region, when there are no more messages on the queue. For example, if five transactions are processed during a single schedule of a program, all five are committed only when the fifth one is completed and the program terminates. Until a transaction has been committed, locks for updated database records are not released and non-express output messages are not queued for output. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing.

If the transaction results in the application calling an external subsystem, such as DB2, the Commit Verify exit provided by the external subsystem can determine whether CMTMODE(MULT) is supported. See documentation under the Commit Verify exit routine in *IMS Version 11 Exit Routines*.

SNGL
Database updates and non-express output messages are committed when the application program completes processing each transaction. IMS invokes commit processing either when the application program

requests the next message (issues a GU to the IO-PCB), or when the application program terminates. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing. If an application ends abnormally before committing its message, emergency restart requeues the message that was in process before the abend and makes it available for reprocessing.

Keyword combination rules include the following:

- CONV(Y) and CMTMODE(MULT) are mutually exclusive.
- WFI(Y) and CMTMODE(MULT) are mutually exclusive.

CONV

Specifies the conversation option.

N The transaction is not conversational.

Y The transaction is conversational. The transaction message is destined for a conversational program. A conversational program processes transactions made up of several steps. A conversational program receives a message from a terminal, replies to the terminal, but saves the data from the transaction in a scratchpad area (SPA). When the person at the terminal enters more data, the program has the data it saved from the last message in the SPA, so it can continue processing the request without the person at the terminal having to enter the data again.

Keyword combination rules include the following:

- CONV(Y) requires SPASZ and SPATRUNC.
- CMTMODE(MULT) and CONV(Y) are mutually exclusive.
- INQ(Y) and CONV(Y) are mutually exclusive.
- RECOVER(N) and CONV(Y) are mutually exclusive.
- SPASZ and CONV(N) are mutually exclusive.
- SPATRUNC and CONV(N) are mutually exclusive.

DCLWA

Specifies the log write-ahead option.

N If N is specified, IMS does not perform log write-ahead. Specify N if input message integrity and the consistency of output messages with associated database updates is not required. DCLWA does not apply to response mode or Fast Path input processing, and is ignored during IMS execution.

Y If Y is specified, IMS performs log write-ahead for recoverable, nonresponse input messages and transaction output messages. This ensures the following:

- A nonresponse input transaction is made recoverable across IMS failures, before IMS acknowledges receipt of the input.
- Database changes are made recoverable before IMS sends associated output reply messages.
- Information in the log buffers is written to the IMS log, before the associated input acknowledgment or output reply is sent to the terminal.

Define DCLWA(Y) for all VTAM terminal types.

DEFAULT(Y)

Specifies this descriptor as the default, which resets the existing default descriptor to DEFAULT(N). When a descriptor is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines transaction descriptor called DFSDSTR1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor. Since only one transaction descriptor can be the default at one time, only one transaction descriptor name can be specified with DEFAULT(Y).

DIRROUTE

Specifies the MSC directed routing option.

N The application program processing a transaction is not informed of the system which originated the transaction. The name of the originating LTERM is placed in the I/O PCB.

Y The application program processing a transaction is informed of the system which originated the transaction, if MSC directed routing is used in a multiple IMS system configuration. An MSNAME corresponding to a logical path back to the originating system is placed in the I/O PCB.

EDITRTN

Specifies the 1- to 8-character name of your transaction input edit routine that edits messages before the program receives the message. This name must be alphanumeric (A through Z, 0 through 9, #, $, and @). It must begin with an alphabetic character (A through Z, #, $, @). It cannot include a blank, comma, period, hyphen, or equal. It cannot include wildcard characters * or %.

The specified edit routine can either be an edit routine defined during the system definition process with the EDIT= parameter on the TRANSACT macro or it can be a new routine. If the routine is a new routine, the routine must reside in one of the RESLIB concatenated data sets. A maximum of 255 input edit routines are supported.

If the edit routine specified on the command is new to IMS, IMS attempts to load the routine from RESLIB. If the load fails, the command is rejected. If the edit routine specified is already defined to IMS, a decision is made whether to load a new copy of the routine or to use the existing copy. If the routine was generated into the system, but there are no transactions referencing the routine, IMS first attempts to load a new copy of the module from RESLIB. If the load is successful, the dynamic copy of the edit routine is used. The generated copy cannot be used again unless IMS is cold started. If the load of the dynamic routine fails, the generated copy is used. If the edit routine was generated into the system and other transactions reference it, the generated copy of the module is used.

EDITRTN is used for a Fast Path potential transaction when the transaction is routed to IMS.

For input from LU 6.2 devices, the user edit exit routine DFSLUEE0 is called instead of the transaction input edit routine specified in EDITRTN.

FP(E) and EDITRTN are mutually exclusive.

A zero value can be specified for the EDITRTN() parm (EDITRTN(0)) to remove an edit routine name from a transaction definition.

EDITUC
Specifies the edit to uppercase option.

N The input data is not translated to uppercase. It can consist of uppercase and lowercase characters as entered from the terminal.

Y The input data is to be translated to uppercase before it is presented to the processing program. If FP(E) or FP(P), the transaction is to be translated to uppercase before being presented to the edit/routing exit routine.

Specifying EDITUC(Y) for VTAM terminals prevents the transmission of embedded device control characters.

EMHBSZ

Specifies the EMH buffer size required to run the Fast Path transaction. This overrides the EMHL execution parameter. If EMHBSZ is not specified, the EMHL execution parameter value is used. The value can be a number from 12 to 30720.

Keyword combination rules include the following:

- EMHBSZ > 0 requires Fast Path to be defined.
- FP(N) and EMHBSZ>0 are mutually exclusive.

EXPRTIME
Specifies the elapsed time in seconds that IMS can use to cancel the input transaction. After a transaction is submitted to IMS, the transaction could be delayed for processing because of a stopped transaction or a potential system slow down. In that case, the user or client application could time out before the transaction is processed. When IMS eventually schedules and processes the transaction, the response message is no longer wanted. With the elapsed time specified for the transaction, IMS can flag the input transaction as expired so that the system does not waste CPU cycles to process the unwanted transaction.

The value can be a number, in seconds, which can range from 0 to 65535. The default is 0, which means that no expiration time is set for this transaction. The transaction expiration attribute is supported by all of the IMS TM interfaces.

Restriction: The transaction expiration checking is not performed at the GU time for Fast Path transactions, IMS conversational transactions, and program-to-program switch transactions.

FP Specifies the Fast Path option.

E The transaction is processed exclusively as Fast Path. The program must be defined as Fast Path exclusive.

N The transaction is not a candidate for Fast Path processing. The program must be defined as not Fast Path. When FP(N) is specified, any attempt to use Fast Path resources or commands might yield unpredictable results.

P The transaction is a potential candidate for Fast Path processing. Fast Path-potential transactions must be able to run under two applications: a Fast Path exclusive application and a non-Fast Path application. A Fast Path exclusive application should be defined to which this

transaction can be routed. Fast Path–potential transactions must be processed by a user edit/routing exit to determine whether the transaction is actually to be processed by IMS Fast Path. If it is to be processed by IMS Fast Path, the edit/routing exit routine associates the transaction with a routing code. This routing code identifies which Fast Path application program is to process the transaction.

The program defined by PGM() must not be defined as Fast Path exclusive.

In order to update a transaction from FP(E) to FP(N) or from FP(N) to FP(E) you must also update the transaction to point to a program that has the same FP() attribute. If you do not update the program attribute, the command fails because of a program conflict.

Keyword combination rules include the following:

- EDITRTN and FP(E) are mutually exclusive.
- EMHBSZ>0 and FP(N) are mutually exclusive.
- FP(E) and FP(P) require Fast Path to be defined.
- MSGTYPE(MULTSEG) and FP(E) are mutually exclusive.
- MSGTYPE(MULTSEG) and FP(P) are mutually exclusive.
- MSNAME and FP(E) are mutually exclusive.
- RECOVER(N) and FP(E) are mutually exclusive.
- RECOVER(N) and FP(P) are mutually exclusive.
- RESP(N) and FP(E) are mutually exclusive.
- RESP(N) and FP(P) are mutually exclusive.
- SIDL and FP(E) are mutually exclusive.
- SIDR and FP(E) are mutually exclusive.

INQ

Specifies the inquiry option.

N This is not an inquiry transaction.

Y This is an inquiry transaction. If INQ(Y) is specified, you can also specify whether this transaction should be recovered during an IMS emergency or normal restart using the RECOVER() parameter.

This option should be specified only for those transactions that, when entered, do not cause a change in any database. Programs are prohibited from issuing ISRT, DLET, or REPL calls to a database when scheduled to process a transaction defined as INQ(Y).

An application program cannot do an SQL INSERT, DELETE, or UPDATE when the IMS transaction is defined with INQ(Y).

Keyword combination rules include the following:

- CONV(Y) and INQ(Y) are mutually exclusive.
- RECOVER(N) and INQ(N) are mutually exclusive.

LCT

Specifies the limit count. This is the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction. The value can be a number from 1 to 65 535. The default is 65 535.

The limit count value is ignored for a transaction processed by a BMP.

The limit count value is ignored in a shared queues environment.

LPRI

Specifies the limit priority. This is the scheduling priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The value can be a number from 0 through 14.

When the limit priority is used and the scheduling priority is raised to the limit priority, the priority is not reduced to the normal priority until all messages enqueued for this transaction name are processed. If you do not want the limit priority for this transaction, define equal values for the normal priority and limit priority, and a limit count of 65535.

When a transaction is processed exclusively by a batch message program (BMP), define the limit priority as 0. If the program specified by PGM() is defined with a program type of batch, the current priority is forced to be 0. However, a batch message processing region (BMP) can process transactions with current scheduling priorities other than 0.

This priority also controls the priority of messages created by this transaction and sent to a destination in a remote system. See also the discussion on MSC priorities under the NPRI definition.

The limit priority value is ignored for a transaction processed by a BMP.

The limit priority value is ignored in a shared-queues environment.

MAXRGN

Specifies the maximum region count. This limits the number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction. When the number of MPP regions is not limited, one transaction might monopolize all available regions. The value can be a number from 0 to 999. MAXRGN(0) means that no limit is imposed. If you define the application program with a scheduling type of SERIAL, omit the MAXRGN parameter or define it as 0.

The following keyword combinations are mutually exclusive:

- PARLIM(65535) and MAXRGN value greater than 0
- SERIAL(Y) and MAXRGN value greater than 0

MSGTYPE

Specifies the message type (single segment or multiple segment). It specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing.

If MSC-directed routing is used in a multiple IMS system configuration, IMS does not ensure that both the message and the transaction destined to process that message are either single segment or multiple segments.

MULTSEG

The incoming message can be more than one segment in length. It is

not eligible for scheduling to an application program until an end-of-message indication is received, or a complete message is created by MFS.

SNGLSEG
The incoming message is one segment in length. It becomes eligible for scheduling when the terminal operator indicates end-of-segment.

Keyword combination rules include the following:

- FP(E) and MSGTYPE(MULTSEG) are mutually exclusive.
- FP(P) and MSGTYPE(MULTSEG) are mutually exclusive.
- RESP(Y) and MSGTYPE(MULTSEG) are mutually exclusive.

MSNAME

Specifies the one- to eight-character name of the logical link path in a multiple IMS system configuration (MSC). A logical link path is a path between any two IMS systems. The IMS systems are identified by the remote system ID and the local system ID associated with the logical link path. The remote system ID identifies the system in which messages using this path are to be processed. The local system ID identifies this system. For an UPDATE TRAN command that is changing a transaction to a remote transaction, or changing the MSC path, the new MSNAME must already be defined.

Keyword combination rules include the following:

- FP(E) and MSNAME are mutually exclusive.
- SIDL and MSNAME are mutually exclusive.
- SIDR and MSNAME are mutually exclusive.

NPRI

Specifies the normal scheduling priority. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The normal priority is assigned to the transaction as the scheduling priority when the number of input transactions enqueued and waiting to be processed is less than the limit count value. The value can be a number from 0 through 14. The default is 1.

This priority also controls the priority of messages created by this transaction and sent to a destination in a remote system.

When a transaction is processed exclusively by a batch message program (BMP), code the normal priority as 0.

When a transaction is processed exclusively by a batch message program (BMP), define the limit priority as 0. If the application program specified by PGM() is defined with a program type of batch, the current priority is forced to be 0. However, a batch message processing region (BMP) can process transactions with current scheduling priorities other than 0.

For remote transactions, the priority used to send the transaction to the processing system, which is termed *the MSC link message priority*. The three MSC link message priority groups are:

- Low
- Medium
- High

The low priority group consists of primary requests in the input terminal system. This group is assigned remote transaction priorities from 0 to 6. The medium group consists of secondary requests, responses, primary requests in an intermediate system, and primary requests in the input terminal system. This group is assigned a remote transaction priority of 7. The high group consists of primary requests in the input terminal system. Messages in this group are assigned remote transaction priorities from 8 to 14. Within each group, messages have a priority based on the current priority value of the transaction or remote transaction in the input terminal system for primary requests, and on the latest processing system for secondary requests and responses.

In an MSC configuration, the transaction priority determines the priority used to send messages inserted by this transaction across an MSC link. If the transaction inserts multiple messages to the same destination (for example, pages to a printer) and these messages must be sent in the order inserted, the normal and limit priority values should be the same. If the normal and limit priority values are not identical, messages inserted at a higher priority than previously inserted messages could arrive at their destination first. (This restriction does not apply to multiple segments of the same message.)

The normal priority value is ignored for a transaction processed by a BMP.

PARLIM

Specifies the parallel processing limit count. This is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. This is the threshold value to be used when the associated application is defined with a scheduling type of parallel. An additional region is scheduled whenever the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.

The value can be a number from 0 to 32767 or 65535. PARLIM(0) indicates that any input message can cause a new region to be scheduled because the scheduling condition is always met (the number of messages is greater than zero). If you specify PARLIM(0), you should specify a MAXRGN value to limit the number of regions that can be scheduled to process a particular transaction. PARLIM(65535) means that parallel processing is disabled and IMS allows the transaction to be scheduled in only one region at a time.

The value specified for PARLIM applies to message processing programs (MPPs) only; it is not supported for batch message processing programs (BMPs).

If you define the application as SERIAL or the scheduling type as SERIAL, define PARLIM(65535).

In a shared-queues environment (when the scheduling type is PARALLEL), any PARLIM value other than 65535 causes a new region to be scheduled whenever the successful consecutive GU count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction. For shared queues environments, the successful consecutive GU count is used instead of the queue count. New regions continue to be scheduled up to the maximum number of regions specified on MAXRGN.

Keyword combination rules include the following:

- MAXRGN>0 and PARLIM(65535) are mutually exclusive.
- SERIAL(Y) and PARLIM value 0 - 32767 are mutually exclusive.

PGM

Specifies the name of the application program associated with the transaction. The program must exist unless the transaction is defined as REMOTE(Y).

PLCT

Specifies the processing limit count. This is maximum number of messages sent to the application program by the IMS for processing without reloading the application program. The value must be a number from 0 through 65535. PLCT(0) means the maximum number of messages sent to the application is one and the application program is reloaded before receiving a subsequent message. PLCT(65535) means that no limit is to be placed upon the number of messages processed at a single program load. Values 1 through 65535 are eligible for quick reschedule processing.

The value is used to determine how many messages an application program is allowed to process in a single scheduling cycle. When the application program requests, and receives, the number of messages indicated, any subsequent requests result in one of two things.

1. IMS indicates “no more messages exist” if any of the following conditions is true
   - The region is not an MPP.
   - The currently scheduled mode is not CMTMODE(SNGL).
   - Equal or higher priority transactions are enqueued for the region.

   IMS might, in fact, have other messages enqueued for the application program. It is the responsibility of the application program to terminate when it receives an indicator that no more messages are available. Termination of the application program makes the region it occupied available for rescheduling. This feature makes it possible for IMS to enable scheduling of higher priority transactions that entered the system while the previous transactions were in process. In addition, if any equal-priority transactions are enqueued, they become eligible for scheduling on a first-in, first-out (FIFO) basis.
2. The region goes through quick reschedule and returns the next message to the application of all of the following conditions are true.
   - The region is an MPP.
   - The transaction is CMTMODE(SNGL).
   - No equal or higher transactions are enqueued.
   - Messages are still enqueued for the application.

PLCTTIME

Specifies the processing limit count time. This is the amount of time (in hundredths of seconds) allowable to process a single transaction (or message). The number specifies the maximum CPU time allowed for each message to be processed in the message processing region.

Batch Message Programs (BMPs) are not affected by this setting.

The value can be a number, in hundredths of seconds, that can range from 1 to 6553500. A value of 6553500 means no time limit is placed on the application program.

If Fast Path is used this specifies, for a given transaction name, the amount of time (in hundredths of seconds) the program is allowed to process a single transaction message. The time represents real time that elapses during transaction processing (not accumulated task time). Real time is used because the input terminal is in response mode and cannot enter another transaction until the response is sent. In this case PLCT() is ignored.

The value controls application program looping. You are not required to optimize the value for program-transaction execution time. However, the time value assigned should not be less than the expected per-transaction execution time. If the scheduled application program exceeds the product of PLCTTIME() and PLCT(), the application program ends abnormally. If an IMS STIMER value of 2 is specified on the DFSMPR procedure, the region does not end abnormally until completion of the DL/I call.

The application program must not use STIMER timer services. IMS uses STIMER timer services to time the execution of transactions. If an application program issues an MVS STIMER macro, it cancels the STIMER timer services set by IMS. Use the STIMERM macro instead for application program timer requests.

RECOVER
Specifies the recovery option.

N The transaction should not be recovered.

Y The transaction should be recovered during an IMS emergency or normal restart.

Keyword combination rules include the following:

- CONV(Y) and RECOVER(N) are mutually exclusive.
- FP(E) and RECOVER(N) are mutually exclusive.
- FP(P) and RECOVER(N) are mutually exclusive.
- INQ(N) and RECOVER(N) are mutually exclusive.

REMOTE
Specifies the remote option.

N The transaction is not remote. The transaction is local and runs on the local system.

Y The transaction is remote. The transaction runs on a remote system.

REMOTE(Y) requires MSNAME or SIDR and SIDL.

RESP
Specifies the response mode option.

N The transaction is not response mode. For terminals specifying or accepting a default of OPTIONS=TRANRESP, input should not stop after this transaction is entered.

Y The transaction is response mode. The terminal from which the transaction is entered is held and prevents further input until a response is received. For terminals specifying or accepting a default of OPTIONS=TRANRESP, no additional messages are to be allowed after this transaction is entered until this transaction sends a response message back to the terminal. Response mode can be forced or negated

by individual terminal definition. RESP(Y) is ignored during online processing for all terminals that do not operate in response mode.

Keyword combination rules include the following:

- FP(E) and RESP(N) are mutually exclusive.
- FP(P) and RESP(N) are mutually exclusive.
- MSGTYPE(MULTSEG) and RESP(Y) are mutually exclusive

SEGNO

Specifies the segment number. This is the maximum number of application program output segments that are allowed into the message queues per Get Unique (GU) call from the application program. The value can be a number from 0 through 65535. If SEGNO(0) is defined, the number of segments is not checked by the online system at execution time.

SEGSZ

Specifies the segment size. This is the maximum number of bytes allowed in any one output segment. The value can be a number from 0 through 65535. If SEGSZ(0) is defined, the segment size is not checked by the online system at execution time.

The maximum output message segment to an LU 6.2 device is 32767. If a transaction is expected to send output to an LU 6.2 device, the SEGSIZE parameter should be no greater than 32767. However, this is not enforced during processing of the command, because IMS cannot determine the device type for the message destination until output time.

SERIAL

Specifies the serial option.

N Messages for the transaction are not processed serially. Message processing can be processed in parallel. Messages are placed on the suspend queue after a U3303 pseudoabend. Scheduling continues until repeated failures result in the transaction being stopped with a USTOP.

Y Messages for the transaction are processed serially. U3303 pseudoabends do not cause the message to be placed on the suspend queue but rather on the front of the transaction message queue, and the transaction is stopped with a USTOP. The USTOP of the transaction is removed when the transaction or the class is started with a /START or UPD TRAN command.

Keyword combination rules include the following:

- MAXRGN>0 and SERIAL(Y) are mutually exclusive.
- PARLIM value 0 - 32767 and SERIAL(Y) are mutually exclusive.

SIDL

Specifies the system identification (SYSID) of the local system in a multiple-IMS system (MSC) configuration. The local system is the originating system to which responses are returned. The value can be a number from 1 to 2036. The local SYSID can be defined in any or all of the MSNAMEs or transactions.

The SIDL parameter is independent of the link type (CTC, MTM, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement.

Keyword combination rules include the following:

- FP(E) and SIDL are mutually exclusive, unless SIDL and SIDR are specified as a pair and are equal to the local system ID of this IMS.
- MSNAME and SIDL are mutually exclusive.
- SIDL value must be defined to this IMS.

SIDR

Specifies the system identification (SYSID) of the remote system in a multiple-IMS system (MSC) configuration. The remote system is the system on which the application program executes. The value can be a number from 1 to 2036. The remote SYSID specified must also be defined for an MSNAME.

The SIDR parameter is independent of the link type (CTC, MTM, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement.

Keyword combination rules include the following:

- FP(E) and SIDR are mutually exclusive, unless SIDL and SIDR are specified as a pair and are equal to the local system ID of this IMS.
- MSNAME and SIDR are mutually exclusive.
- SIDR value must be defined to this IMS.

SPASZ

Specifies the scratchpad area (SPA) size, in bytes, for a conversational transaction. The value can be a number from 16 and 32767.

Keyword combination rules include the following:

- CONV(N) and SPASZ are mutually exclusive.

SPATRUNC

Specifies the scratchpad area (SPA) truncation option of a conversational transaction. This defines whether the SPA data should be truncated or preserved across a program switch to a transaction that is defined with a smaller SPA.

When a conversation initially starts, and when a program is switched, the SPATRUNC option is checked and set or reset as specified. When the option is set, it remains set for the life of the conversation, or until a program switch occurs to a transaction that specifies the option is to be Reset.

When a program switch occurs, the truncated data option for the new transaction is first checked, and that specification is set for the conversation and is used for the SPA inserted into the output message. If the option is not specified for the new transaction, the option currently in effect for the conversation is used.

S IMS preserves all of the data in the SPA, even when a program switch is made to a transaction that is defined with a smaller SPA. The transaction with the smaller SPA does not see the truncated data, but when the transaction switches to a transaction with a larger SPA, the truncated data is used.

R The truncated data is not preserved.

Keyword combination rules include the following:

- CONV(N) and SPATRUNC are mutually exclusive.

TRANSTAT

Specifies whether transaction level statistics should be logged for message driven programs. If Y is specified, transaction level statistics are written to the log in a X'56FA' log record.

N Transaction level statistics should not be logged.

Y Transaction level statistics should be logged.

The TRANSTAT keyword on the UPDATE TRAN or UPDATE TRANDESC command gives the user the ability to override the system default or the current value of the TRANSTAT parameter. If the TRANSTAT keyword is omitted on the UPDATE TRAN or UPDATE TRANDESC command, the current transaction level statistics setting is unchanged.

WFI

Specifies the wait-for input option. This attribute does not apply to Fast Path transactions, which always behave as wait-for-input transactions.

N This is not a wait-for-input transaction.

Y This is a wait-for-input transaction. A message processing or batch processing application program that processes WFI transactions are scheduled and invoked normally. If the transaction to be processed is defined as WFI, the program is allowed to remain in main storage after it has processed the available input messages. The QC status code (no more messages) is returned to the program if the PROCLIM count (PLCT) is reached; a command is entered to change the status of the scheduled transaction, database, program, or class; commands relating to the databases used by the transaction are entered, or IMS is terminated with a checkpoint shutdown.

Keyword combination rules include the following:

- MODE(MULT) and WFI(Y) are mutually exclusive.

Usage notes

If all the attributes specified by the UPDATE command are already defined for the descriptor, no update is actually made, no descriptors are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

Descriptors exist for the life of the IMS unless they are deleted using a DELETE command. The descriptors are recoverable across an IMS warm start or emergency restart. Descriptors are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up. Each descriptor is updated individually. Individual updating does not work like online change where either all descriptors are updated or no descriptors are updated.

The UPDATE TRANDESC command can be issued only through the OM API. This command applies to DB/DC and DCCTL systems. This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE TRANDESC commands are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined). This command is recoverable. If the descriptor being updated is the IMS-defined transaction descriptor (DFSDSTR1) the only attribute that can be changed is the DEFAULT attribute.

Output fields

The following table shows the UPDATE TRANDESC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 410. Output fields for the UPDATE TRANDESC command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DESC	TRANDESC	Transaction descriptor name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a non-zero completion code and contains information that further explains the completion code.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 411. Return and reason codes for the UPDATE TRANDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000004'	X'00002008'	Invalid number of keywords. A SET keyword is required.
X'00000008'	X'00002100'	CMTMODE(MULT) mutually exclusive with WFI(Y).
X'00000008'	X'00002101'	CONV(Y) mutually exclusive with CMTMODE(MULT).
X'00000008'	X'00002102'	CONV(Y) mutually exclusive with INQ(Y).
X'00000008'	X'00002103'	CONV(N) mutually exclusive with SPASZ>0 and SPATRUNC.
X'00000008'	X'00002104'	CONV(Y) mutually exclusive with RECOVER(N).
X'00000008'	X'00002105'	CONV(Y) requires SPASZ and SPATRUNC.
X'00000008'	X'00002108'	Invalid EDITRTN name.
X'00000008'	X'0000210C'	FP(E) mutually exclusive with EDITRTN.

Table 411. Return and reason codes for the UPDATE TRANDESC command (continued)

Return code	Reason code	Meaning
X'00000008'	X'0000210E'	FP(E) or FP(P) mutually exclusive with MSC keyword MSNAME or SIDR and SIDL.
X'00000008'	X'0000210F'	FP(E) or FP(P) mutually exclusive with MSGTYPE(MULTSEG).
X'00000008'	X'00002110'	FP(N) mutually exclusive with EMHBSZ > 0.
X'00000008'	X'00002111'	FP(E) or FP(P) mutually exclusive with RECOVER(N).
X'00000008'	X'00002112'	FP(E) or FP(P) mutually exclusive with RESP(N).
X'00000008'	X'00002116'	INQ(N) mutually exclusive with RECOVER(N).
X'00000008'	X'00002119'	MSC keyword MSNAME or SIDL/SIDR mutually exclusive with application program defined as Fast Path exclusive (FP(E)) associated with this transaction.
X'00000008'	X'0000211A'	Invalid MSNAME name.
X'00000008'	X'0000211B'	MSNAME mutually exclusive with SIDL and SIDR.
X'00000008'	X'0000211D'	MAXRGN>0 mutually exclusive with PARLIM(65535).
X'00000008'	X'0000211E'	MAXRGN>0 mutually exclusive with SERIAL(Y).
X'00000008'	X'00002121'	PARLIM value mutually exclusive with SERIAL(Y).
X'00000008'	X'00002125'	REMOTE(Y) requires MSC keyword MSNAME or SIDR and SIDL.
X'00000008'	X'00002126'	Invalid SIDL value.
X'00000008'	X'00002127'	SIDL/SIDR must be specified as a pair. Either SIDL was specified alone or SIDR was specified alone.
X'00000008'	X'00002128'	Invalid SIDL value.
X'00000008'	X'00002133'	Multiple name parameters were specified with DEFAULT(Y). Only one descriptor can be the default at one time.
X'0000000C'	X'00003000'	Command was successful for some of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), output lines are only returned for descriptors with non-zero completion codes. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined, FP(E), FP(P), or EMHBSZ >0 invalid.
X'00000010'	X'00004120'	Online change phase is in progress.

Table 411. Return and reason codes for the UPDATE TRANDESC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004310'	Storage could not be obtained for the Transaction Input edit routine table. A cold start is required to fix this error.
X'00000010'	X'00004314'	The Transaction Input edit routine could not be loaded.
X'00000010'	X'00004318'	A new Transaction Input edit routine could not be added. The maximum of 255 routines has already been reached.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.
X'00000014'	X'00005010'	Latch could not be obtained.
X'00000014'	X'000050FF'	The UPDATE TRANDESC command processing terminated because of an internal error.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 412. Completion codes for the UPDATE TRANDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for transaction descriptor.
10	NO RESOURCES FOUND	Transaction descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this transaction descriptor. This could also mean this command, if the descriptor is specified by more than one specific or wildcard parameter.
19	CMTMODE(MULT)/ WFI(Y) CONFLICT	Transaction descriptor update failed because commit mode multiple CMTMODE(MULT) option conflicts with the wait-for-input WFI(Y) option.
1B	CONV(Y)/ CMTMODE(MULT) CONFLICT	Transaction descriptor update failed because the conversation CONV(Y) option conflicts with the commit mode multiple CMTMODE(MULT) option.
1D	CONV(Y)/INQ(Y) CONFLICT	Transaction descriptor update failed because the conversation CONV(Y) option conflicts with the inquiry only INQ(Y) option.
1E	CONV(N)/SPASZ OR SPATRUNC CONFLICT	Transaction descriptor update failed because the CONV(N) option conflicts with the SPA size or SPA truncation option.

Table 412. Completion codes for the UPDATE TRANDESC command (continued)

Completion code	Completion code text	Meaning
1F	CONV(Y)/ RECOVER(N) CONFLICT	Transaction descriptor update failed because the conversation CONV(Y) option conflicts with the nonrecoverable RECOVER(N) option.
27	NOT ALLOWED FOR DEFAULT DESCRIPTOR	Transaction descriptor update failed because the default descriptor cannot update any runtime resource definition attribute.
2F	FP(E)/ BMPTYPE(Y) CONFLICT	Transaction descriptor update failed, because Fast Path exclusive FP(E) option conflicts with program defined as batch BMPTYPE(Y).
35	FP(E)/EDITRTN CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) option conflicts with the edit routine EDITRTN.
36	FP(E)/FP(N) PGM CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) option conflicts with program defined as non-Fast Path FP(N).
38	NOT ALLOWED FOR A BMP	The UPDATE TRAN command is invalid for the resource because the PSB associated with the transaction descriptor is a BMP.
3A	FP(E OR P)/MSC KEYWORD CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with the MSC MSNAME, SIDR/SIDL option.
3B	FP(E OR P)/ MSGTYPE(MULTSEG) CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with message type multiple segment MSGTYPE(MULTSEG) option.
3C	MAXRGN>0/ PARLIM(65535) CONFLICT	Transaction descriptor update failed because maximum region count MAXRGN value conflicts with the parallel limit count PARLIM value 65535, which means parallel scheduling is disabled. MAXRGN > 0 is not allowed with PARLIM(65535).
3E	FP(N)/FP(E) PGM CONFLICT	Transaction descriptor update failed because non-Fast Path FP(N) option conflicts with program defined as Fast Path exclusive FP(E).
3F	FP(P)/BMPTYPE(Y) CONFLICT	Transaction descriptor update failed because Fast Path potential program FP(P) conflicts with the program defined as batch BMPTYPE(Y).
40	PARLIM/ SCHDTYPE(SERIAL) CONFLICT	The PARLIM cannot be changed for the resource because the PSB associated with the transaction is defined as does not have parallel scheduling.
41	FP(E OR P)/ RECOVER(N) CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with nonrecoverable RECOVER(N) option.
42	FP(E OR P)/RESP(N) CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with response mode RESP(N) option.

Table 412. Completion codes for the UPDATE TRANDESC command (continued)

Completion code	Completion code text	Meaning
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for the IMS descriptor or resource. DFSDSTR1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
49	INQ(N)/ RECOVER(N) CONFLICT	Transaction descriptor update failed because non-inquiry INQ(N) option conflicts with nonrecoverable RECOVER(N) option.
4F	INVALID MAXRGN VALUE	Maximum region MAXRGN value is invalid.
5E	MAXRGN>0/ SERIAL(Y) CONFLICT	Transaction descriptor update failed because non-zero maximum region value conflicts with serial SERIAL(Y) option.
61	DFSBCB STORAGE ERROR.	DFSBCB storage could not be obtained.
6B	PARLIM/SERIAL(Y) CONFLICT	Transaction descriptor update failed because the parallel limit PARLIM value conflicts with the serial SERIAL(Y) option.
6D	INVALID PROGRAM NAME	
79	REMOTE(Y) REQUIRES MSC KEYWORD	Transaction descriptor update failed because remote REMOTE(Y) option requires MSC keyword such as MSNAME or SIDR/SIDL.
8A	WILDCARD PARAMETER NOT SUPPORTED	A wildcard parameter was specified with DEFAULT(Y). Only one descriptor can be the default at one time.

Examples

The following are examples of the UPDATE TRANDESC command:

Example 1 for UPDATE TRANDESC command

TSO SPOC input:

```
UPD TRANDESC NAME(*) SET(FP(E)) OPTION(ALLRSP)
```

TSO SPOC output:

```
Response for: UPD TRANDESC NAME(*) SET(FP(E)) OPTION(ALLRSP)
DescName MbrName    CC CCText
CONVDESC IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DESC001  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DESC002  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DESC003  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DESC004  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DESC005  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DESC101  IMS1        0
DESC102  IMS1        0
DESC103  IMS1        0
DESC104  IMS1        0
DESC105  IMS1        0
DESC201  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DESC202  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DESC203  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
```

```
DESC204  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DESC205  IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
DFSDSTR1 IMS1       3B FP/MSGTYPE(MULTSEG) CONFLICT
FPEDESC  IMS1        0
FPPDESC  IMS1       36 FP(E)/FP(N) PGM CONFLICT
MSCDESC  IMS1       3A FP(E OR P)/MSC KEYWORD CONFLICT
```

OM API input:

```
CMD(UPDATE TRANDESC NAME(*) SET(OPTION(ALLRSP)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.312 22:26:57.314733</statime>
<stotime>2006.312 22:26:57.315449</stotime>
<staseq>BFAD419D67BADF8C</staseq>
<stoseq>BFAD419D67E79D0C</stoseq>
<rqsttkn1>USRT011 10142657</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>TRANDESC        </kwd>
<input>UPD TRANDESC NAME(*) SET(FP(E)) OPTION(ALLRSP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="GBL" llbl="Global" scope="GBL" sort="d" key="2" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="y" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="CONVID" llbl="ConvID" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="NODE" llbl="NodeName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="USER" llbl="UserName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="LU" llbl="LUName" scope="LCL" sort="n" key="0"
 scroll="yes" len="24" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMEM" llbl="TMember" scope="LCL" sort="n" key="0"
scroll="yes" len="16" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TPIP" llbl="TPipe" scope="LCL" sort="n" key="0" scroll="yes"
```

```
 len="8" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DESC102 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC004 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(DESC205 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(DFSDSTR1) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(DESC201 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(FPPDESC ) MBR(IMS1) CC(  36) CCTXT(FP(E)/FP(N) PGM CONFLICT)
</rsp>
<rsp>DESC(DESC103 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC005 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(DESC001 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(DESC202 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(DESC104 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(CONVDESC) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(DESC002 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(DESC203 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(MSCDESC ) MBR(IMS1) CC(  3A) CCTXT(FP(E OR P)/MSC KEYWORD
 CONFLICT) </rsp>
<rsp>DESC(DESC105 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC101 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC003 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
<rsp>DESC(DESC204 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE(MULTSEG)
 CONFLICT) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE TRANDESC command is issued to update all transaction descriptors to be Fast Path exclusive. The update succeeded for some transaction descriptors and failed for others. The update succeeded for some transaction descriptors as shown by the completion code 0. The update failed for transaction descriptor FPPDESC with completion code 36, which indicates that the FP(E) attribute conflicts with the FP(N) attribute defined for the program referred to by FPPDESC. The update failed for transaction descriptor MSCDESC with completion code 3A, which indicates that the FP(E) attribute conflicts with the MSC settings already defined for descriptor MSCDESC. The update failed for some transaction descriptors with completion code 3B, which indicates that the FP(E) attribute conflicts with the MSGTYPE(MULTSEG) already defined for the transaction.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

Commit Verify exit routine (Exit Routines)

Chapter 30. /VUNLOAD command

The /VUNLOAD AREA command removes the specified areas from the z/OS data space or coupling facility.

All of the updated CIs for the area are written to DASD. All subsequent I/O for the area is from DASD. /VUNLOAD processing occurs concurrently with application processing.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Example" on page 854

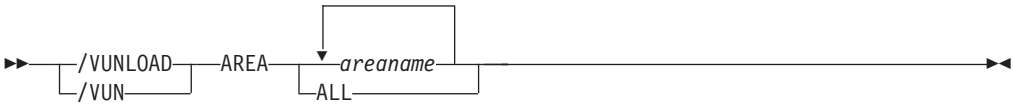
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command and keyword can be issued.

Table 413. Valid environments for the /VUNLOAD command and keyword

Command / Keyword	DB/DC	DBCTL	DCCTL
/VUNLOAD	X	X	
AREA	X	X	

Syntax



Usage notes

In a data-sharing environment, in order to maintain data integrity, IMS requests IRLM to lock the first control interval (CI) in exclusive mode before IMS unloads the area from the coupling facility. This lock causes those IMS subsystems sharing the area to complete any synchronization point processing. When all sharing subsystems are no longer using the area, IMS requests IRLM to release the CI locks and IMS unloads the area. All subsequent I/O for the area is from DASD.

The /VUNLOAD command does not change any of the VSO options set in the RECON data set. Therefore, at the next IMS restart or /START AREA command, the VSO options again take effect. Any changes to VSO definitions must be made through DBRC commands. For Shared VSO areas however, the /VUNLOAD command is persistent across IMS restarts and can be reset only by a /START AREA command. This is true for shared VSO areas for consistency across all sharing partners. This prevents one system from accessing data from DASD while others are accessing from the coupling facility.

© Copyright IBM Corp. 1974, 2015

For multiple VSO DEDB areas that share a single coupling facility structure, when the /VUNLOAD AREA command is issued, one of the following actions may occur:

- If there is only one area in the structure (either a single-area structure or a multi-area structure with one area), the area is taken out of VSO and the structure is deleted. The IMS issuing the /VUNLOAD AREA command notifies the other IMS systems sharing the area to disconnect from the structure.
- If there are multiple areas in the structure, when one area is taken out of VSO, the area is disassociated and its CIs are deleted in the structure. The IMS issuing the /VUNLOAD AREA notifies the other IMS systems sharing the area to either disassociate or disconnect from the structure depending on if the IMS is still sharing or not sharing other areas in the structure.

The /VUNLOAD command is not valid for an RSR tracking subsystem.

This command can be issued to an IMSplex using the Batch SPOC utility.

Example

Entry ET:

```
/VUNLOAD
```

Response ET:

```
NC0000000 FPEC 03070 09:58:03.52 01   00000290  R 14,/VUN AREA DD01AR0
NR8400000 FPEC 03070 09:58:03.52 JOB00116 00000090  IEE600I REPLY TO 14 IS;/VUN AREA DD01AR0
N 8400000 FPEC 03070 09:58:03.53 JOB00116 00000090  DFS058I 09:58:03 VUNLOAD COMMAND IN PROGRESS   SYS3
W 8400000 FPEC 03070 09:58:03.53 JOB00116 00000090 *17 DFS996I *IMS READY*  SYS3
N 8400000 FPEC 03070 09:58:04.42 JOB00116 00000090  DFS2823I AREA DD01AR0
 DISCONNECT FROM STR: DD01AR0STR1 SUCCESSFULS                                                SYS3
N 8400000 FPEC 03070 09:58:04.42 JOB00116 00000090  DFS0488I VUN COMMAND COMPLETED. AREA= DD01AR0  RC= 0 SYS3
```

Notices

This information was developed for products and services offered in the U.S.A.

This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

© Copyright IBM Corp. 1974, 2015

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to

IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS," without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Privacy policy considerations

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, See IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

Bibliography

This bibliography lists all of the publications in the IMS Version 11 library, supplemental publications, publication collections, and accessibility titles cited in the IMS Version 11 library.

IMS Version 11 library

Title	Acronym	Order number
IMS Version 11 Application Programming	APG	SC19-2428
IMS Version 11 Application Programming APIs	APR	SC19-2429
IMS Version 11 Commands, Volume 1: IMS Commands A-M	CR1	SC19-2430
IMS Version 11 Commands, Volume 2: IMS Commands N-V	CR2	SC19-2431
IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands	CR3	SC19-2432
IMS Version 11 Communications and Connections	CCG	SC19-2433
IMS Version 11 Database Administration	DAG	SC19-2434
IMS Version 11 Database Utilities	DUR	SC19-2435
IMS Version 11 Diagnosis	DGR	GC19-2436
IMS Version 11 Exit Routines	ERR	SC19-2437
IMS Version 11 Installation	INS	GC19-2438
IMS Version 11 Licensed Program Specifications	LPS	GC19-2439
IMS Version 11 Master Index and Glossary	MIG	SC19-2440
IMS Version 11 Messages and Codes, Volume 1: DFS Messages	MC1	GC19-4232
IMS Version 11 Messages and Codes, Volume 2: Non-DFS Messages	MC2	GC19-4233
IMS Version 11 Messages and Codes, Volume 3: IMS Abend Codes	MC3	GC19-4234
IMS Version 11 Messages and Codes, Volume 4: IMS Component Codes	MC4	GC19-4235
IMS Version 11 Operations and Automation	OAG	SC19-2441
IMS Version 11 Release Planning	RPG	GC19-2442
IMS Version 11 System Administration	SAG	SC19-2443
IMS Version 11 System Definition	SDG	GC19-2444
IMS Version 11 System Programming APIs	SPR	SC19-2445
IMS Version 11 System Utilities	SUR	SC19-2446

Supplementary publications

Title	Order number
IMS and SOA Executive Overview	GC19-2516
Program Directory for Information Management System Transaction and Database Servers V11.0	GI10-8788
IMS Version 11 Fact Sheet	GC19-2451
IRLM Messages and Codes	GC19-2666

© Copyright IBM Corp. 1974, 2015

Publication collections

Title	Format	Order number
IMS Version 11 Product Kit	CD	SK5T-7375

Accessibility titles cited in the IMS Version 11 library

Title	Order number
z/OS TSO/E Primer	SA22-7787
z/OS TSO/E User's Guide	SA22-7794
z/OS ISPF User's Guide Volume 1	SC34-4822

Index

Special characters

/NRESTART command
  description 1
  environment 1
  examples 7
  keywords 3
  syntax 1
  usage notes 6
/OPNDST command
  description 11
  environment 11
  examples 15
  keywords 12
  syntax 11
  usage notes 14
/PSTOP command
  description 19
  environment 19
  examples 23
  keywords 20
  syntax 19
  usage notes 23
/PURGE command
  description 29
  environment 29
  examples 32
  keywords 30
  syntax 29
  usage notes 31
/QUIESCE command
  description 383, 384
  environment 383
  example 384
  keywords 383
  syntax 383
/RCLSDST command
  description 385
  environment 385
  example 386
  syntax 385
/RCOMPT command
  description 387
  environment 387
  example 388
  syntax 387
  usage notes 388
/RDISPLAY command
  environment 389
  examples 390
  keywords 389
  syntax 389
  usage notes 389
/RECOVER command 391
  ADD keyword
    RCVTOKEN 393
    USEAREA 393
  START
    ERRORABORT keyword 402
/RELEASE command
  description 411
  environment 411
/RELEASE command *(continued)*
  example 412
  keywords 411
  syntax 411
  usage notes 411
/RESET command
  description 413
  environment 413
  example 413
  syntax 413
/RMCHANGE command
  usage notes 417
/RMDELETE command
  usage notes 417
/RMGENJCL command
  usage notes 417
/RMINIT command
  usage notes 417
/RMLIST command
  usage notes 417
/RMNOTIFY command
  usage notes 417
/RMxxxxxx command
  DBRC modifiers 416
  description 415
  environment 415
  examples 418
  keywords 416
  LTERM keyword 416
  parameters passed to DBRC 416
  syntax 416
  usage notes 417
/RSTART command
  description 427, 430
  environment 427
  examples 430
  keywords 428
  syntax 427
/RTAKEOVER command
  description 433
  environments 433
  examples 435
  syntax 433
  usage notes 433
/SECURE command
  description 437
  environment 437
  examples 439
  keywords 438
  syntax 437
/SET command
  description 441
  environment 441
  examples 443
  keywords 441
  syntax 441
  usage notes 442
/SIGN command
  description 445
  environment 445
  examples 448
/SIGN command *(continued)*
  keywords 445
  syntax 445
  usage notes 448
/SMCOPY command
  description 451
  environment 451
  example 452
  keywords 451
  syntax 451
  usage notes 452
/SSR command
  description 453
  environment 453
  example 453
  syntax 453
  usage notes 453
/START command
  APPC keyword
    description 456
    environment 456
    syntax 456
  AREA keyword
    description 456
    environment 456
    examples 459
    keywords 457
    syntax 457
    usage notes 458
  AUTARCH keyword
    environment 460
  AUTOARCH keyword
    description 460
    examples 460
    syntax 460
  CLASS keyword
    description 461
    environment 461
    examples 461
    syntax 461
  DATAGRP keyword
    description 462
    environment 462
    keywords 462
    syntax 462
    usage notes 464
  DB keyword
    description 465
    environment 465
    examples 471
    keywords 466
    syntax 465
    usage notes 469
  DC keyword
    description 473
    environment 473
    syntax 474
  DESC keyword
    description 474
    environment 474
    syntax 474

© Copyright IBM Corp. 1974, 2015

/START command *(continued)*
description 455
ISOLOG keyword
description 474
environment 475
syntax 475
LINE keyword
description 475
environment 475
examples 476
syntax 475
usage notes 476
LTERM keyword
description 477
environment 477
examples 478
syntax 478
usage notes 478
LUNAME keyword
description 478
environment 479
syntax 479
usage notes 479
MADSIOT keyword
description 480
environment 480
syntax 480
usage notes 480
MSNAME keyword
description 481
environment 481
examples 481
syntax 481
NODE keyword
description 482
environment 482
examples 483
syntax 482
usage notes 482
OLDS keyword
description 483
environment 484
examples 484
syntax 484
OTMA keyword
description 484
environment 484
examples 485
syntax 485
usage notes 485
PGM keyword
description 485
environment 485
examples 486
syntax 486
REGION keyword
description 487
environment 487
examples 488
syntax 487
usage notes 487
RTC keyword
description 489
environment 489
examples 490
syntax 489
/START command *(continued)*
SB keyword
description 490
environment 490
examples 491
syntax 490
SERVGRP keyword
description 491
environment 491
examples 492
syntax 491
usage notes 492
SLDSREAD keyword
description 492
environment 492
syntax 493
SUBSYS keyword
description 493
environment 493
examples 494
keywords 493
syntax 493
usage notes 494
SURV keyword
description 494
environment 494
syntax 495
usage notes 495
THREAD keyword
description 495
environment 495
syntax 495
usage notes 496
TMEM keyword
description 496
environment 496
examples 498
keywords 496
syntax 496
TRAN keyword
description 498
environment 498
examples 500
syntax 499
usage notes 499
TRKARCH keyword
environment 501
syntax 501
TRKASRCH keyword
description 501
USER keyword
description 501
environment 502
examples 503
syntax 502
usage notes 502
VGR keyword
description 503
environment 503
keywords 504
syntax 504
WADS keyword
description 504
environment 504
syntax 504
XRCTRACK keyword
description 505
/START command *(continued)*
XRCTRACK keyword *(continued)*
environment 505
syntax 505
/STOP command
ADS keyword
description 508
environment 508
syntax 508
usage notes 508
APPC keyword
description 508
environment 509
keywords 509
syntax 509
usage notes 509
AREA keyword
description 509
environment 510
examples 512
keywords 510
syntax 510
usage notes 512
AUTOARCH keyword
description 513
environment 513
examples 513
syntax 513
BACKUP keyword
description 513
environment 514
syntax 514
CLASS keyword
description 514
environment 514
examples 515
syntax 514
usage notes 514
DATAGRP keyword
description 515
environment 515
syntax 516
usage notes 516
DB keyword
description 516
environment 516
examples 519
syntax 517
usage notes 517
DC keyword
description 520
environment 520
syntax 521
usage notes 521
DESC keyword
description 521
environment 521
syntax 521
description 507
LINE keyword
description 522
environment 522
examples 522
syntax 522
usage notes 522
LTERM keyword
description 523

/STOP command *(continued)*
LTERM keyword *(continued)*
environment 524
examples 524
syntax 524
usage notes 524
LUNAME keyword
description 525
environment 525
syntax 525
usage notes 525
MADSIOT keyword
description 526
environment 526
syntax 526
usage notes 526
MSNAME keyword
description 527
environment 527
examples 527
syntax 527
NODE keyword
description 528
environment 528
examples 529
syntax 528
usage notes 528
OLDS keyword
description 530
environment 530
examples 530
syntax 530
usage notes 530
OTMA keyword
description 531
environment 531
examples 531
syntax 531
usage notes 531
PGM keyword
description 532
environment 532
examples 532
syntax 532
usage notes 532
REGION keyword
description 533
environment 533
examples 535
keywords 534
reg# parameter 534
reg#-#reg parameter 534
syntax 533
RTC keyword
description 541
environment 541
syntax 541
SB keyword
description 541
environment 542
examples 542
syntax 542
SERVGRP keyword
description 542
environment 543
syntax 543
usage notes 543
/STOP command *(continued)*
SLDSREAD keyword
description 543
environment 543
syntax 544
SUBSYS keyword
description 544
environment 544
examples 545
syntax 544
usage notes 544
SURV keyword
description 545
environment 546
keywords 546
syntax 546
THREAD keyword
description 546
environment 546
examples 547
keywords 547
syntax 547
usage notes 547
TMEM keyword
description 549
environment 549
examples 550
keywords 550
syntax 549
usage notes 550
TRAN keyword
description 551
environment 551
examples 552
syntax 551
usage notes 551
USER keyword
description 553
environment 553
examples 554
syntax 553
usage notes 553
VGR keyword
description 555
environment 555
syntax 555
usage notes 555
WADS keyword
description 555
environment 556
syntax 556
XRCTRACK keyword
description 556
environment 556
syntax 556
/SWITCH command 557
description 557
environment 557
examples 559
keywords 558
/TERMINATE OLREORG command
command response 577
examples 578
type-1 577, 578
/TEST command 581
description 581
environment 581
/TEST command *(continued)*
examples 583
keywords 581
usage notes 582
/TRACE command
description 585
DFSMSCE0 keyword 586
examples 589
EXIT keyword
description 586
environment 586
keywords 586
syntax 586
usage notes 586
LEVEL keyword 588, 590, 597, 617
LINE keyword
description 587
environment 587
examples 589
keywords 588
syntax 587
LINK keyword
description 589
environment 589
examples 591
keywords 590
syntax 590
LUNAME keyword
description 592
environment 592
syntax 592
usage notes 592
MODULE keyword 588, 591, 598, 617
MONITOR keyword
description 593
environment 593
examples 596
keywords 594
parameter environments table 595
syntax 593
usage notes 595
NODE keyword
description 596
environment 596
examples 599
keywords 597
syntax 597
usage notes 598
OSAMGTF keyword
description 600
environment 600
syntax 600
PGM keyword
description 603
environment 603
syntax 603
usage notes 603
PI keyword
description 600
environment 601
examples 602
keywords 601
syntax 601
usage notes 602
PSB keyword
description 605

/TRACE command *(continued)*
PSB keyword *(continued)*
environment 605
examples 606
keywords 605
syntax 605
usage notes 605
TABLE keyword
description 606
environment 606
examples 609
keywords 606
syntax 606
usage notes 609
TAKEOVER keyword 589, 591, 598
TCO keyword
description 610
environment 611
syntax 611
TIMEOUT keyword
description 611
environment 611
keywords 612
syntax 611
usage notes 612
TMEMBER keyword
description 612
environment 613
examples 614
keywords 613
syntax 613
usage notes 613
TPIPE keyword 613
TRAN keyword
description 614
environment 614
examples 615
syntax 614
usage notes 614
TRAP keyword
description 615
environment 616
syntax 616
UNITYPE keyword
description 616
environment 616
keywords 617
syntax 616
usage notes 618
/UNLOCK command
DB keyword
description 620
environment 620
examples 621
syntax 620
usage notes 620
LTERM keyword
description 621
environment 622
syntax 622
usage notes 622
NODE keyword
description 622
environment 622
syntax 622
PGM keyword
description 623
/UNLOCK command *(continued)*
PGM keyword *(continued)*
environment 623
examples 623
syntax 623
PTERM keyword
description 624
environment 624
examples 625
syntax 624
SYSTEM keyword
description 625
environment 625
examples 626
syntax 625
usage notes 625
TRAN keyword
description 627
environment 627
examples 627
syntax 627
usage notes 627
/VUNLOAD command
description 853
environment 853
syntax 853
usage notes 853

A

ABDUMP keyword
SWITCH command 558
ACCESS keyword
START command 462, 466
accessibility
features x
keyboard shortcuts x
ACTIVE keyword
SWITCH command 558
active system
master terminal display screen 560
ALLENTRIES keyword
RECOVER command
REMOVE keyword 398
STOP keyword 406
alternate system
master terminal display screen 559
AOI application programs
UNLOCK DB command 621
UNLOCK PGM command 623
UNLOCK TRAN command 627
APPC keyword
PURGE command 30
SECURE command 438
APPL keyword
SIGN command 447
AREA keyword
RECOVER command 394
REMOVE keyword 398
STOP keyword 406

B

BACKUP keyword
SWITCH command 558
BISYNC link
resetting continuous mode 20
BUILDQ keyword
NRESTART command 3

C

CAGROUP keyword
RECOVER command
ADD 394
REMOVE keyword 398
STOP keyword 406
CANCEL keyword
STOP command 509, 535
CHECK parameter
SECURE command 438
CHECKPOINT keyword
NRESTART command 4
SWITCH command 558
CMDAUTH keyword
NRESTART command 4
CMDAUTHE keyword
NRESTART command 4
command
NRESTART 1
OPNDST command 11
PSTOP command 19
PURGE 29
QUIESCE command 383
RCLSDST command 385
RCOMPT command 387
RDISPLAY command 389
RECOVER 391
RELEASE command 411
RESET command 413
RMCHANGE command 415
RMDELETE command 415
RMGENJCL command 415
RMINIT command 415
RMLIST command 415
RMNOTIFY command 415
RMxxxxxx command 415
RSTART command 427
RTAKEOVER 433
SECURE 437
SET command 441
SIGN 445
SMCOPY command 451
SSR command 453
START command 455
STOP command 507
SWITCH command 557
TEST command 581
TRACE 585
UNLOCK command 619
UPDATE 629
VUNLOAD 853
commands
prerequisite knowledge vii
control block
trace
information 588, 590, 597, 617
module 588, 591, 598, 617
CONVERSATION keyword
RELEASE command 411
SET command 441

CTC (channel-to-channel)
  link
    RSTART command 428

D

data sharing
  NRESTART command 4
database recovery control (DBRC) 415
DATAGROUP keyword
  RECOVER command
    REMOVE keyword 398
DB keyword
  RECOVER command
    ADD 394
    REMOVE keyword 398
    STOP keyword 406
DBALLOC keyword
  START command 464, 467
DBDS keyword
  RECOVER command
    ADD 394
    REMOVE keyword 398
    STOP keyword 407
DBDSGRP keyword
  RECOVER command
    ADD 394
    REMOVE keyword 398
    STOP keyword 407
DBRC (database recovery control)
  modifiers
    RMxxxxxx command 416
  RMCHANGE command 415, 417
  RMDELETE command 417
  RMGENJCL command 417
  RMINIT command 417
  RMLIST command 417
  RMNOTIFY command 417
  RMxxxxxx command 415
DFSMSCE0 keyword
  /TRACE command 586
display screen
  active system 560
  alternate system 559
  EEQE (extended error queue element) 626
  I/O toleration 626
  UNLOCK SYSTEM 626

E

EEQE (extended error queue element)
  display screen 626
ERRORABORT keyword
  RECOVER command
    START 402
ERRORCONT keyword
  RECOVER command
    START 402
example
  OM API
    TERMINATE OLREORG 579
examples
  /TRACE command 589
  QUERY STRUCTURE command 286
  QUERY TRAN command 313
examples *(continued)*
  TERMINATE OLREORG command 579

F

Fast Path
  exclusive transactions
    PSTOP command 23
  message-driven programs
    PURGE command 30
  potential transactions
    PSTOP command 23
  region
    STOP REGION command 534
  reset terminal response mode 428
  RSTART command 428
  START command 482
FORCE keyword
  PSTOP command 20
  SWITCH command 558
FORMAT keyword
  NRESTART command 4
FPPROG keyword
  PURGE command 31
FPRGN keyword
  PURGE command 31
FULL parameter
  SECURE command 438

G

GLOBAL keyword
  START command 457, 468
  STOP command 510
GROUP keyword
  NEWPW command 447
  SIGN command 447

H

HIDAM database
  starting 470

I

I/O toleration
  display screen 626
ID keyword
  OPNDST command 12
IMS keyword 693
ISC (Intersystem Communication)
  node
    shutdown and deallocation 384
  RCOMPT command
    valid parameters 388
  STOP NODE USER command 528

J

JOIN parameter
  SECURE command 439

K

keyboard shortcuts x

L

legal notices
  notices 855
  trademarks 855
LEVEL keyword
  /TRACE command 588, 590, 597, 617
LINE keyword
  PSTOP command 20
  PURGE command 31
  RSTART command 428
  TEST command 582
LINK keyword
  PSTOP command 20
  RSTART command 428
LOCAL keyword
  START command 458, 464, 468
  STOP command 511
LOGOND keyword
  OPNDST command 12
LOPEN keyword
  RSTART command 428
LTERM keyword
  PSTOP command 21
  PURGE command 31
  RMxxxxxx command 416
  SET command 442
LU 6.2 device
  releasing a conversation 411

M

MASTER parameter
  RDISPLAY command 389
  SMCOPY command 451
master terminal
  UNLOCK DB command 621
  UNLOCK PGM command 623
  UNLOCK TRAN command 627
MFS keyword
  TEST command 581
MFSTEST mode
  TEST command 582
MODE keyword
  OPNDST command 12
  RSTART command 429
MODULE keyword
  /TRACE command 588, 591, 598, 617
MONITOR keyword
  /TRACE command
    parameter environments table 595
MPP (message processing program)
  stopping processing within a specified region 534
MRQ (Message Requeuer program)
  default MRQ BMP program name 603
  NRESTART BUILDQ command fails 3
MSDB (main storage database)
  checkpoint data set 5

MSDB (main storage database) *(continued)*
NRESTART command 5
STOP DATABASE command 517
MSDBLOAD keyword
NRESTART command 5
MSNAME keyword
PURGE command 31
MSPLINK keyword
PSTOP command 21
RSTART command 429
MTM
link
RSTART command 428
MULTSIGN keyword
NRESTART command 5
MVS/ESA
STOP JES2 CANCEL command 535
STOP MVS/ESA CANCEL command 535

N

NOBACKOUT keyword
START command 468
NOBUILDQ keyword
NRESTART command 3
NOCHECK keyword
RECOVER command
START 402
NOCMDAUTH keyword
NRESTART command 5
NOCMDAUTHE keyword
NRESTART command 5
NODBALLOC keyword
START command 463, 467
NODE keyword
OPNDST command 13
QUIESCE command 383
RSTART command 429
TEST command 582
NONE parameter
SECURE command 438, 439
NOPFA keyword
STOP command 511
NOTRANAUTH keyword
NRESTART command 5
NOUSER keyword
NRESTART command 5
NRESTART command
BUILDQ keyword 3
CHECKPOINT keyword 4
CMDAUTH keyword 4
CMDAUTHE keyword 4
description 1
environment 1
examples 7
FORMAT keyword 4
keywords 3
MSDBLOAD keyword 5
MULTSIGN keyword 5
NOBUILDQ keyword 3
NOCMDAUTH keyword 5
NOCMDAUTHE keyword 5
NOTRANAUTH keyword 5
NOUSER keyword 5
NRESTART command *(continued)*
restart
security definition 6
SNGLSIGN keyword 6
syntax 1
TRANAUTH keyword 6
usage notes 6
USER keyword 6
with data sharing 4
nuserpw keyword
SIGN command 447, 448

O

OFF keyword
SIGN command 446
OFFLINE keyword
RECOVER command
ADD 392
START 402
OLC keyword 563
OLDS keyword
SWITCH command 558
OLREORG keyword 576
ON parameter
SIGN command 445
OPNDST command
description 11
environment 11
examples 15
ID keyword 12
keywords 12
LOGOND keyword 12
MODE keyword 12
NODE keyword 13
Q keyword 13
syntax 11
UDATA keyword 14
usage notes 14
USER keyword 14
USERD keyword 14
OTMA keyword
SECURE command 438

P

parameters
system initialization, displayed 6, 455
PassTicket keyword
SIGN command 448
performance
generating data with /TRACE 602
PGMDESC keyword 249
PITR keyword
RECOVER command
START 402
preset mode
RCLSDST command 385
resetting
/IAM command 442
SET command 442
START command 482
PROCLIB library
STOP SUBSYS command 545
PROFILE parameter
SECURE command 438, 439
PSTOP command
description 19
environment 19
equivalent IMS type-2 commands 23
examples 23
FORCE keyword 20
keywords 20
LINE keyword 20
LINK keyword 20
LTERM keyword 21
MSPLINK keyword 21
PURGE keyword 20
REGION keyword 22
syntax 19
TRAN keyword 22
usage notes 23
PTERM keyword
TEST command 582
publications
changes xi
organization xi
titles xi
PURGE command
APPC keyword 30
description 29
environment 29
equivalent IMS type-2 commands 31
examples 32
FPPROG keyword 31
FPRGN keyword 31
keywords 30
LINE keyword 31
LTERM keyword 31
MSNAME keyword 31
syntax 29
TRAN keyword 31
usage notes 31
PURGE keyword
PSTOP command 20

Q

Q keyword
OPNDST command 13
QUERY AREA command
parameters 39
QUERY command 249, 318
AREA keyword
completion codes 43
description 36
environment 36
examples 44
keywords 37
output fields 41
reason codes 43
return codes 43
similar IMS commands 41
status conditions 40
syntax 36
usage notes 39
DB keyword
command comparison 55
completion codes 63, 64
description 49
environment 49

QUERY command *(continued)*
DB keyword *(continued)*
examples 64
keywords 50
NAME() 50
output fields 55
reason codes 63
return codes 63
SHOW() 50
similar to IMS commands 55
status conditions 60
STATUS() 53
syntax 49
usage notes 54
DBDESC keyword
completion codes 76
description 71
environment 71
examples 77
keywords 72
output fields 73
reason codes 76
return codes 76
syntax 72
usage notes 73
environments
TRAN keyword 287
IMS keyword
completion codes 82
description 79
environment 79
examples 83
keywords 79
output fields 80
reason codes 82
return codes 82
syntax 79
usage notes 80
IMSPLEX keyword
completion codes 89
description 84
environment 84
examples 90
keywords 84
member subtypes 87
member types 87
output fields 88
reason codes 89
return codes 89
status conditions 86
syntax 84
usage notes 86
LE keyword
completion codes 95
description 93
environment 93
examples 96
keywords 94
output fields 95
reason codes 95
return codes 95
syntax 93
usage notes 95
LTERM keyword
command comparison 108
completion codes 112
description 100
QUERY command *(continued)*
LTERM keyword *(continued)*
environment 100
examples 114
keywords 102
output fields 108
reason codes 112
return codes 112
similar to IMS commands 108
status 111
syntax 101
usage notes 107
MEMBER keyword
completion codes 125
description 119
environment 119
examples 125
keywords 119
output fields 124
reason codes 125
return codes 125
syntax 119
usage notes 120
MSLINK keyword
completion codes 129
description 129
environment 129
examples 129
keywords 129
output fields 129
reason codes 129
return codes 129
syntax 129
usage notes 129
MSNAME keyword
completion codes 148
description 144
environment 144
example 150
keywords 145
output fields 147
reason codes 148
return codes 148
syntax 144
usage notes 147
MSPLINK keyword
completion codes 154
description 150
environment 151
examples 155
keywords 151
output fields 153
reason codes 154
return codes 154
syntax 151
usage notes 153
NODE keyword
command comparison 165
completion codes 173
description 156
environment 157
examples 175
keywords 159
output fields 166
reason codes 173
return codes 173
similar to IMS commands 165
QUERY command *(continued)*
NODE keyword *(continued)*
status 171
syntax 157
usage notes 164
ODBM keyword
description 180
ODBM TYPE(ALIAS) keyword
completion codes 181
description 180
environment 180
example 182
keywords 180
output fields 181
reason codes 181
return codes 181
syntax 180
usage notes 181
ODBM TYPE(CONFIG) keyword
completion codes 185
description 183
environment 183
example 186
keywords 183
output fields 184
reason codes 185
return codes 185
syntax 183
usage notes 184
ODBM TYPE(DATASTORE) keyword
completion codes 190
description 187
environment 187
example 190
keywords 188
output fields 189
reason codes 190
return codes 190
syntax 187
usage notes 189
ODBM TYPE(SCIMEMBER) keyword
completion codes 193
description 191
environment 192
example 194
keywords 192
output fields 192
reason codes 193
return codes 193
syntax 192
usage notes 192
ODBM TYPE(THREAD) keyword
completion codes 198
description 194
environment 194
example 199
keywords 195
output fields 197
reason codes 198
return codes 198
syntax 194
usage notes 197
ODBM TYPE(TRACE) keyword
completion codes 203
description 202
environment 202
example 204

QUERY command *(continued)*
ODBM TYPE(TRACE) keyword *(continued)*
keywords 202
output fields 203
reason codes 203
return codes 203
syntax 202
usage notes 203
OLC keyword
completion codes 208
description 204
environment 205
examples 209
keywords 205
output fields 206
reason codes 208
return codes 208
syntax 205
usage notes 206
OLREORG keyword
completion codes 217
description 213
environment 214
examples 218
keywords 214
output fields 216
reason codes 217
return codes 217
syntax 214
usage notes 216
OTMADESC keyword
completion codes 224
description 220
environment 220
examples 224
keywords 221
output fields 223
reason codes 224
return codes 224
syntax 220
usage notes 222
OTMATI keyword
completion codes 231
description 226
environment 226
examples 232
keywords 227
output fields 230
reason codes 231
return codes 231
syntax 226
usage notes 229
PGM keyword
completion codes 244
description 233
environment 234
examples 245
keywords 235
output fields 238
reason codes 244
return codes 244
syntax 234
usage notes 237
PGMDESC keyword
completion codes 255
description 249
QUERY command *(continued)*
PGMDESC keyword *(continued)*
environment 249
examples 256
keywords 250
output fields 252
reason codes 255
return codes 255
usage notes 251
POOL keyword
completion codes 262
description 258
environment 259
examples 263
keywords 259
output fields 260
reason codes 262
return codes 262
syntax 259
usage notes 259
RTC keyword
completion codes 273
description 268
environment 268
examples 274
keywords 269
output fields 271
reason codes 273
return codes 273
syntax 268
usage notes 271
RTCDESC keyword
completion codes 280
description 276
environment 277
examples 281
keywords 277
output fields 279
reason codes 280
return codes 280
syntax 277
usage notes 278
STRUCTURE keyword
completion codes 285
description 283
environment 283
keywords 283
output fields 284
reason codes 285
return codes 285
syntax 283
usage notes 284
TRAN keyword
completion codes 310
description 287
environment 287
output 299
reason codes 310
return codes 310
similar to other IMS commands 298
syntax 288
usage notes 298
TRANDESC keyword
completion codes 331
description 318
environment 318
QUERY command *(continued)*
TRANDESC keyword *(continued)*
examples 332
keywords 319
output fields 324
reason codes 331
return codes 331
usage notes 324
USER keyword
command comparison 342
completion codes 348
description 336
environment 336
examples 350
keywords 337
output fields 342
reason codes 348
return codes 348
similar to IMS commands 342
status 347
syntax 336
usage notes 341
USEREXIT keyword
completion codes 357
description 354
environment 354
examples 358
keywords 355
output fields 356
reason codes 357
return codes 357
syntax 354
usage notes 356
USERID keyword
command comparison 364
completion codes 366
description 360
environment 361
examples 367
keywords 361
output fields 364
reason codes 366
return codes 366
similar to IMS commands 364
status 365
syntax 361
usage notes 363
QUERY DB command
parameters 50
QUERY LE command
keywords 94
QUERY MEMBER command
attributes 120
status conditions 121
QUERY OLC command
parameters 206
QUERY OLREORG command
parameters 214
QUERY STRUCTURE command
examples 286
QUERY TRAN command
examples 313
parameters 289
QUEUE command
LTERM keyword
completion codes 374
description 371, 372

QUEUE command *(continued)*
LTERM keyword *(continued)*
environment 371
examples 375
output fields 373
reason codes 374
return codes 374
syntax 371
usage notes 373
TRAN keyword
completion codes 379
description 376
environment 376
example 381
keywords 377
output fields 379
reason codes 379
return codes 379
syntax 377
usage notes 378
QUIESCE command
description 383, 384
environment 383
example 384
keywords 383
NODE keyword 383
syntax 383
USER keyword 383

R

RACF (Resource Access Control facility)
SIGN ON command 445
RCF= parameter
overriding with NRESTART COLDSYS command 7
RCLSDST command
description 385
environment 385
example 386
syntax 385
RCOMPT command
description 387
environment 387
example 388
syntax 387
usage notes 388
RCVTIME keyword
RECOVER command
START 402
RCVTOKEN keyword
/RECOVER command
ADD 393
RECOVER command
REMOVE keyword 397
START 401
RDISPLAY command
environment 389
equivalent IMS type-2 commands 389
examples 390
keywords 389
MASTER parameter 389
syntax 389
usage notes 389
READNUM keyword
RECOVER command
START 402
RECOVER command 391
ADD keyword
AREA 394
CAGROUP 394
DB 394
DBDS keyword 394
DBDSGRP keyword 394
description 391
environment 391
examples 395
keywords 392
OFFLINE keyword 392
RCVTOKEN 393
RECOVGRP keyword 394
STAGLOBAL keyword 393
STALOCAL keyword 393
syntax 392
usage notes 395
USEAREA 393
USEDBDS keyword 393
REMOVE keyword
ALLENTRIES keyword 398
AREA keyword 398
CAGROUP keyword 398
DATAGROUP keyword 398
DB keyword 398
DBDS keyword 398
DBDSGRP keyword 398
description 396
environment 396
examples 399
keywords 397
RCVTOKEN keyword 397
RECOVGRP keyword 398
syntax 397
usage notes 399
START keyword
description 400
environment 401
ERRORABORT 402
ERRORCONT 402
examples 403
keywords 401
NOCHECK keyword 402
OFFLINE keyword 402
PITR keyword 402
RCVTIME 402
RCVTOKEN 401
READNUM 402
STAGLOBA 402
STALOCAL 402
syntax 401
usage notes 403
STOP keyword
ALLENTRIES keyword 406
AREA keyword 406
CAGROUP keyword 406
DB keyword 406
DBDS keyword 407
DBDSGRP keyword 407
description 405
environment 405
examples 407
keywords 406
RECOVER command *(continued)*
STOP keyword *(continued)*
RECOVGRP keyword 407
SAVE keyword 406
syntax 406
usage notes 407
TERMINATE keyword
description 408
environment 408
examples 409
synax 408
usage notes 409
RECOVGRP keyword
RECOVER command
ADD keyword 394
REMOVE keyword 398
STOP keyword 407
reg# parameter
/STOP command
REGION keyword 534
reg#-#reg parameter
/STOP command
REGION keyword 534
REGION keyword
/STOP command
reg# parameter 534
reg#-#reg parameter 534
PSTOP command 22
RELEASE command
CONVERSATION keyword 411
description 411
environment 411
example 412
keywords 411
LU 6.2 device 411
syntax 411
usage notes 411
RESET command
description 413
environment 413
example 413
syntax 413
RMxxxxxx command
description 415
keywords 416
syntax 416
RS parameter
NRESTART command 4
RSTART command
description 427, 430
environment 427
equivalent IMS type-2 commands 430
examples 430
keywords 428
LINE keyword 428
LINK keyword 428
LOPEN keyword 428
MODE keyword 429
MSPLINK keyword 429
NODE keyword 429
syntax 427
USER keyword 429
RTAKEOVER command
active subsystem 434
description 433
environments 433

RTAKEOVER command *(continued)*
examples 435
syntax 433
tracking subsystem 434
usage notes 433

S

SAVE keyword
RECOVER command
STOP keyword 406
SECURE command 439, 440
APPC keyword 438
CHECK parameter 438
description 437
environment 437
examples 439
FULL parameter 438
JOIN parameter 439
keywords 438
NONE parameter 438, 439
OTMA keyword 438
PROFILE parameter 438, 439
syntax 437
security
definition
at cold start 6
SET command
CONVERSATION keyword 441
description 441
environment 441
examples 443
keywords 441
LTERM keyword 442
syntax 441
TRANSACTION keyword 442
usage notes 442
SGN= parameter
overriding with NRESTART
COLDSYS command 7
SIGN command
APPL keyword 447
description 445
environment 445
examples 448
GROUP keyword 447
keywords 445
NEWPW keyword 447
nuserpw keyword 447, 448
OFF keyword 446
ON parameter 445
PassTicket keyword 448
syntax 445
usage notes 448
USERD keyword 447
userpw keyword 447
VERIFY keyword 448
signon
RACF 445
terminals requiring
commands accepted 445
SMCOPY command
description 451
environment 451
example 452
keywords 451
MASTER parameter 451
SMCOPY command *(continued)*
syntax 451
TERMINAL keyword 452
usage notes 452
SNGLSIGN keyword
NRESTART command 6
SSM keyword
START command 493
SSR command
description 453
environment 453
example 453
syntax 453
usage notes 453
STAGLOBAL keyword
RECOVER command
ADD keyword 393
START 402
STALOCAL keyword
RECOVER command
ADD keyword 393
START 402
START command
ACCESS keyword 462, 466
APPC keyword
description 456
environment 456
syntax 456
AREA keyword
description 456
environment 456
equivalent IMS type-2
commands 459
examples 459
keywords 457
syntax 457
usage notes 458
AUTARCH keyword
environment 460
AUTOARCH keyword
description 460
examples 460
keywords 460
syntax 460
CLASS keyword
description 461
environment 461
examples 461
syntax 461
DATABASE keyword
DBALLOC keyword 464
LOCAL keyword 464
NODBALLOC keyword 463
DATAGRP keyword
description 462
environment 462
equivalent IMS type-2
commands 464
keywords 462
syntax 462
usage notes 464
DB keyword
DBALLOC keyword 467
description 465
environment 465
equivalent IMS type-2
commands 471
START command *(continued)*
DB keyword *(continued)*
examples 471
keywords 466
NODBALLOC keyword 467
syntax 465
usage notes 469
DC keyword
description 473
environment 473
syntax 474
DESC keyword
description 474
environment 474
syntax 474
description 455
GLOBAL keyword 457, 468
ISOLOG keyword
description 474
environment 475
syntax 475
LINE keyword
description 475
environment 475
examples 476
syntax 475
usage notes 476
LOCAL keyword 458, 468
LTERM keyword
description 477
environment 477
examples 478
syntax 478
usage notes 478
LUNAME keyword
description 478
environment 479
keywords 479
syntax 479
usage notes 479
MADSIOT keyword
description 480
environment 480
syntax 480
usage notes 480
MSNAME keyword
description 481
environment 481
examples 481
syntax 481
NOBACKOUT keyword 468
NODE keyword
description 482
environment 482
examples 483
syntax 482
usage notes 482
OLDS keyword
description 483
environment 484
examples 484
syntax 484
OTMA keyword
description 484
environment 484
examples 485
syntax 485

START command *(continued)*
OTMA keyword *(continued)*
usage notes 485
PGM keyword
description 485
environment 485
equivalent IMS type-2 commands 486
examples 486
syntax 486
REGION keyword
description 487
environment 487
examples 488
syntax 487
usage notes 487
RTC keyword
description 489
environment 489
equivalent IMS type-2 commands 489
examples 490
syntax 489
SB keyword
description 490
environment 490
examples 491
syntax 490
SERVGRP keyword
description 491
environment 491
examples 492
syntax 491
usage notes 492
SLDSREAD keyword
description 492
environment 492
syntax 493
SSM keyword 493
SUBSYS keyword
description 493
environment 493
examples 494
keywords 493
syntax 493
usage notes 494
SURV keyword
description 494
environment 494
syntax 495
usage notes 495
THREAD keyword
description 495
environment 495
syntax 495
usage notes 496
TMEM keyword
description 496
environment 496
examples 498
keywords 496
syntax 496
TPIPE keyword 497
TRAN keyword
description 498
environment 498
START command *(continued)*
TRAN keyword *(continued)*
equivalent IMS type-2 commands 500
examples 500
syntax 499
usage notes 499
TRKARCH keyword
description 501
environment 501
syntax 501
USER keyword
description 501
environment 502
examples 503
syntax 502
usage notes 502
VGR keyword
description 503
environment 503
keywords 504
syntax 504
WADS keyword
description 504
environment 504
syntax 504
XRCTRACK keyword
description 505
environment 505
syntax 505
STOP command
ADS keyword
description 508
environment 508
syntax 508
usage notes 508
APPC keyword
description 508
environment 509
keywords 509
syntax 509
usage notes 509
AREA keyword
description 509
environment 510
equivalent IMS type-2 commands 512
examples 512
keywords 510
syntax 510
usage notes 512
AUTOARCH keyword
description 513
environment 513
examples 513
syntax 513
BACKUP keyword
description 513
environment 514
syntax 514
CANCEL keyword 509, 535
CLASS keyword
description 514
environment 514
examples 515
syntax 514
usage notes 514
STOP command *(continued)*
DATAGRP keyword
description 515
environment 515
equivalent IMS type-2 commands 516
syntax 516
usage notes 516
DB keyword
description 516
environment 516
equivalent IMS type-2 commands 519
examples 519
syntax 517
usage notes 517
DC keyword
description 520
environment 520
syntax 521
usage notes 521
DESC keyword
description 521
environment 521
syntax 521
description 507
GLOBAL keyword 510
JES2 CANCEL 535
LINE keyword
description 522
environment 522
examples 522
syntax 522
usage notes 522
LOCAL keyword 511
LTERM keyword
description 523
environment 524
examples 524
syntax 524
usage notes 524
LUNAME keyword
description 525
environment 525
syntax 525
usage notes 525
MADSIOT keyword
description 526
environment 526
syntax 526
usage notes 526
MSNAME keyword
description 527
environment 527
examples 527
syntax 527
MVS/ESA CANCEL 535
NODE keyword
description 528
environment 528
examples 529
syntax 528
usage notes 528
NOPFA keyword 511
OLDS keyword
description 530
environment 530

STOP command *(continued)*
OLDS keyword *(continued)*
examples 530
syntax 530
usage notes 530
OTMA keyword
description 531
environment 531
examples 531
syntax 531
usage notes 531
PGM keyword
description 532
environment 532
equivalent IMS type-2 commands 532
examples 532
syntax 532
usage notes 532
REGION keyword
description 533
environment 533
examples 535
keywords 534
syntax 533
REGION TRANSACTION keywords
stopping WFI mode 534
RTC keyword
description 541
environment 541
equivalent IMS type-2 commands 541
syntax 541
SB keyword
description 541
environment 542
examples 542
syntax 542
SERVGRP keyword
description 542
environment 543
syntax 543
usage notes 543
SLDSREAD keyword
description 543
environment 543
syntax 544
SUBSYS keyword
description 544
environment 544
examples 545
syntax 544
usage notes 544
SURV keyword
description 545
environment 546
keywords 546
syntax 546
THREAD keyword
description 546
environment 546
examples 547
keywords 547
syntax 547
usage notes 547
TMEM keyword
description 549
STOP command *(continued)*
TMEM keyword *(continued)*
environment 549
examples 550
keywords 550
syntax 549
usage notes 550
TPIPE keyword
OTMA 550
TRAN keyword
description 551
environment 551
equivalent IMS type-2 commands 552
examples 552
syntax 551
usage notes 551
USER keyword
description 553
environment 553
examples 554
syntax 553
usage notes 553
VGR keyword
description 555
environment 555
syntax 555
usage notes 555
WADS keyword
description 555
environment 556
syntax 556
XRCTRACK keyword
description 556
environment 556
syntax 556
SWITCH command 557
ABDUMP keyword 558
ACTIVE keyword 558
BACKUP keyword 558
CHECKPOINT keyword 558
description 557
environment 557
examples 559
FORCE keyword 558
keywords 558
OLDS keyword 558
SYSTEM keyword 558
WADS keyword 558
syntax 249, 318, 557, 563, 576, 581, 693
syntax diagram
how to read viii
system console
UNLOCK DB command 621
UNLOCK PGM command 623
UNLOCK TRAN command 627
system initialization parameters, displayed 6, 455
SYSTEM keyword
SWITCH command 558

T

TAKEOVER keyword
/TRACE command 589, 591, 598
TERMINAL keyword
SMCOPY command 452
TERMINATE command 563, 576
OLC keyword
completion codes 567
description 563
environment 563
example 574
output fields 566
reason codes 567
return codes 567
usage notes 563
OLREORG keyword
completion codes 578
description 575
example 578
keywords 576
output fields 577
reason codes 578
return codes 578
usage notes 577
TERMINATE OLC command
error handling 564
output fields 566
TERMINATE OLREORG command
environments 575
examples 579
TEST command 581
description 581
environment 581
examples 583
keywords 581
LINE keyword 582
MFS keyword 581
NODE keyword 582
PTERM keyword 582
usage notes 582
USER keyword 582
TPIPE keyword
/TRACE command 613
START command 497
STOP command 550
TRACE command
description 585
DFSMSCE0 keyword 586
examples 589
EXIT keyword
description 586
environment 586
keywords 586
syntax 586
usage notes 586
LEVEL keyword 588, 590, 597, 617
LINE keyword
description 587
environment 587
examples 589
keywords 588
syntax 587
LINK keyword
description 589
environment 589
examples 591
keywords 590
syntax 590
LUNAME keyword
description 592
environment 592
syntax 592

TRACE command *(continued)*
LUNAME keyword *(continued)*
usage notes 592
MODULE keyword 588, 591, 598, 617
MONITOR keyword
description 593
environment 593
examples 596
keywords 594
parameter environments table 595
syntax 593
usage notes 595
NODE keyword
description 596
environment 596
examples 599
keywords 597
syntax 597
usage notes 598
OSAMGTF keyword
description 600
environment 600
syntax 600
PGM keyword
description 603
environment 603
equivalent IMS type-2 commands 604
syntax 603
usage notes 603
PI keyword
description 600
environment 601
examples 602
keywords 601
syntax 601
usage notes 602
PSB keyword
description 605
environment 605
examples 606
keywords 605
syntax 605
usage notes 605
TABLE keyword
description 606
environment 606
examples 609
keywords 606
syntax 606
usage notes 609
TAKEOVER keyword 589, 591, 598
TCO keyword
description 610
environment 611
syntax 611
TIMEOUT keyword
description 611
environment 611
keywords 612
syntax 611
usage notes 612
TMEMBER keyword
description 612
environment 613
examples 614
TRACE command *(continued)*
TMEMBER keyword *(continued)*
keywords 613
syntax 613
usage notes 613
TPIPE keyword 613
TRAN keyword
description 614
environment 614
equivalent IMS type-2 commands 615
examples 615
syntax 614
usage notes 614
TRAP keyword
description 615
environment 616
syntax 616
UNITYPE keyword
description 616
environment 616
keywords 617
syntax 616
usage notes 618
trademarks 855
TRAN keyword
PSTOP command 22
PURGE command 31
TRANAUTH keyword
NRESTART command 6
TRANDESC keyword 318
transaction authorization
terminals requiring signon 445
TRANSACTION keyword
SET command 442
TRN= parameter
overriding with NRESTART COLDSYS command 7
type-1 commands
/TERMINATE OLREORG
command response 577
examples 578
type-2 commands
QUERY 35
QUEUE 371
TERMINATE 563
UPDATE 629

U

UDATA keyword
OPNDST command 14
UNLOCK command
DB keyword
description 620
environment 620
equivalent IMS type-2 commands 621
examples 621
syntax 620
usage notes 620
entered from AOI application programs 621, 623, 627
entered from master terminal 619
entered from system console 619
LTERM keyword
description 621
UNLOCK command *(continued)*
LTERM keyword *(continued)*
environment 622
syntax 622
usage notes 622
NODE keyword
description 622
environment 622
syntax 622
PGM keyword
description 623
environment 623
equivalent IMS type-2 commands 623
examples 623
syntax 623
PTERM keyword
description 624
environment 624
examples 625
syntax 624
SYSTEM keyword
description 625
display screen format 626
environment 625
examples 626
syntax 625
usage notes 625
TRAN keyword
description 627
environment 627
equivalent IMS type-2 commands 628
examples 627
syntax 627
usage notes 627
UPDATE AREA command
parameters 631
similar IMS commands 635
UPDATE command 693
AREA keyword
completion codes 636, 637
description 629
environment 630
examples 640
keywords 631
output fields 635
reason codes 636
return and reason codes 636
return codes 636
similar IMS commands 635
syntax 630
usage notes 634
DATAGRP keyword
completion codes 649
description 643
environment 643
examples 656
keywords 644
output fields 648, 649
reason codes 649
return codes 649
similar IMS commands 648
syntax 644
usage notes 647
DB keyword
completion codes 673, 674

UPDATE command *(continued)*
DB keyword *(continued)*
description 660
environment 660
equivalent IMS commands 672
examples 683
keywords 661
output fields 672
reason codes 673
return and reason codes 673
return codes 673
syntax 660
usage notes 670
DBDESC keyword
completion codes 689
description 686
environment 686
examples 691
keywords 687
output fields 689
reason codes 689
return codes 689
syntax 686
IMS keyword
completion codes 697
description 693
environment 693
examples 699
keywords 693
output fields 697
reason codes 697
return codes 697
usage notes 696
LE keyword
completion codes 704
description 701
environment 702
examples 705
keywords 702
output fields 703
reason codes 704
return codes 704
syntax 702
usage notes 703
MSLINK keyword
completion codes 714
description 707
environment 707
examples 716
keywords 708
output fields 714
reason codes 714
return codes 714
syntax 707
usage notes 713
MSNAME keyword
completion codes 721
description 717
environment 717
examples 723
keywords 718
output fields 720
reason codes 721
return codes 721
syntax 717
MSPLINK keyword
completion codes 727
UPDATE command *(continued)*
MSPLINK keyword *(continued)*
description 724
environment 724
examples 728
keywords 725
output fields 727
reason codes 727
return codes 727
syntax 725
usage notes 726
ODBM keyword
description 729
ODBM START(CONNECTION) keyword
completion codes 731
description 729
environment 730
example 732
keywords 730
output fields 731
reason codes 731
return codes 731
syntax 730
usage notes 730
ODBM START(TRACE) keyword
completion codes 734
description 733
environment 733
example 735
keywords 733
output fields 733
reason codes 734
return codes 734
syntax 733
usage notes 733
ODBM STOP(CONNECTION) keyword
completion codes 737
description 735
environment 736
example 738
keywords 736
output fields 737
reason codes 737
return codes 737
syntax 736
usage notes 736
ODBM STOP(TRACE) keyword
completion codes 740
description 739
environment 739
example 741
keywords 739
output fields 739
reason codes 740
return codes 740
syntax 739
usage notes 739
ODBM TYPE(CONFIG) keyword
completion codes 743
description 741
environment 741
example 744
keywords 741
output fields 742
reason codes 743
UPDATE command *(continued)*
ODBM TYPE(CONFIG) keyword *(continued)*
return codes 743
syntax 741
usage notes 742
OLREORG keyword
command responses 747
completion codes 748
description 745
environment 745
examples 749
keywords 746
output fields 747
reason codes 748
return codes 748
syntax 746
OTMADESC keyword
completion codes 753
description 750
environment 750
examples 754
keywords 751
output fields 752
reason codes 753
return codes 753
syntax 751
usage notes 752
PGM keyword
completion codes 763
description 755
environment 756
examples 767
keywords 757
output fields 763
reason codes 763
return codes 763
syntax 756
usage notes 762
PGMDESC keyword
completion codes 775
description 769
environment 769
examples 777
keywords 770
output fields 774
reason codes 775
return codes 775
syntax 770
usage notes 774
POOL keyword
completion codes 781
description 779
environment 779
examples 781
keywords 779
output fields 780
reason codes 781
return codes 781
syntax 779
usage notes 780
RTC keyword
completion codes 785
description 782
environment 783
examples 787
keywords 783

UPDATE command *(continued)*
RTC keyword *(continued)*
output fields 785
reason codes 785
return codes 785
syntax 783
usage notes 784
RTCDESC keyword
completion codes 792
description 789
environment 789
examples 793
keywords 790
output fields 791
reason codes 792
return codes 792
syntax 789
usage notes 791
TRAN keyword
completion codes 820, 823
description 795
environment 795
equivalent IMS commands 818
examples 826
keywords 798
output fields 819
reason codes 820
return codes 820
syntax 795
usage notes 544, 817
TRANDESC keyword
completion codes 845
description 829
environment 829
examples 849
keywords 831
output fields 845
reason codes 845
return codes 845
syntax 829
usage notes 844
type-2 629
UPDATE DATAGRP command
similar IMS commands 648
UPDATE DB command
equivalent IMS commands 672
parameters 661
UPDATE LE command
parameters 702
UPDATE OLREORG command
examples 749
UPDATE TRAN command
equivalent IMS commands 818
parameters 798
USEAREA keyword
/RECOVER command
ADD 393
USEDBDS keyword
RECOVER command
ADD keyword 393
USER keyword
NRESTART command 6
OPNDST command 14
QUIESCE command 383
RSTART command 429
TEST command 582
USERD keyword
OPNDST command 14
SIGN command 447
userpw keyword
SIGN command 447

V

VTAM (Virtual Telecommunications Access Method)
link
RSTART command 428
MSC links
PSTOP MSPLINK command 21
terminal
disconnecting 385
initiating a session 14
ready/not ready state 388
VUNLOAD command
description 853
entry format 853
environment 853
syntax 853
usage notes 853

W

WADS keyword
SWITCH command 558
wait-for-input mode
stopping message processing 534

X

XRF (Extended Recovery Facility)
alternate system
START REGION command 488

Product Number: 5635-A02

Printed in USA

SC19-2431-05

Spine information:

IMS **Version 11**

Commands, Volume 2: IMS Commands N - V