IBM z/VSE
Version 6 Release 1

VSE/VSAM Commands

SC34-2707-01

Note !

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 313.

This edition applies to Version 6 Release 1 of IBM® z/Virtual Storage Extended (z/VSE), Program Number 5686-VS6, and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC34–2707–00.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the addresses given below.

A form for readers' comments is provided at the back of this publication. If the form has been removed, address your comments to:

```
IBM Deutschland Research & Development GmbH
Department 3282
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany
```

You may also send your comments by FAX or via the Internet:

```
Internet: s390id@de.ibm.com
FAX (Germany): 07031-16-3456
FAX (other countries): (+49)+7031-16-3456
```

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures ..... **ix**

Tables ..... **xi**

About This Publication ..... **xiii**
Who Should Use This Publication xiii
How to Use This Publication xiii
Where to Find More Information xiii

Summary of Changes ..... **xv**

Chapter 1. The IDCAMS Utility Program **1**
Functions of IDCAMS 1
Types of IDCAMS Commands 3
Functional Commands 3
Modal Commands 4
Coding of IDCAMS Commands 4
Understanding Syntax Diagrams 4
Syntax of IDCAMS Commands 6
Password Requirements for IDCAMS Commands 9
Job Control 11
IDCAMS Input and Output Files 11
Tape Considerations for IDCAMS 12
Device Dependencies for IDCAMS 14
Defining Objects in a Catalog 15
Defining a Catalog 16
Defining a VSE/VSAM Data Space 19
Distributed Free Space 21
Defining a VSE/VSAM File (Cluster) 22
Loading Records into a File 31
Alternate Indexes 32
Defining a Path 35
Defining a NonVSAM File 37
Defining Work Files on Virtual Disk 37
Altering Catalog Entries 38
Specifying Information That Alters an Entry 38
Deleting Catalog Entries 39
Specifying Information that Deletes an Entry 39
Using REPRO for Catalog Backup and File Reorganization 40
Backing Up a Catalog 40
Unloading a Catalog 40
Reloading a Catalog 41
Using EXPORT/IMPORT for Transporting or Backing Up Files 44
EXPORT: Making a File Portable 46
IMPORT: Loading a Portable File 46
Listing Catalog Entries 47
Printing Data Records 47
Verifying a File's Accessibility 48
Using BACKUP and RESTORE 48

Functions ... 49
Advantages of BACKUP and RESTORE over EXPORT and IMPORT ... 51
Performance Considerations when Changing Files During Restore ... 52
When to Use EXPORT/IMPORT and when BACKUP/RESTORE ... 53
The BACKUP Command (Job Control) ... 53
The RESTORE Command (Job Control) ... 54
IDCAMS Interactive Interface (IDCONS) ... 55
Overview ... 55
Syntax and Usage ... 56
Example ... 56

Chapter 2. The IDCAMS Commands ... 59
Functional Commands and Formats ... 59
ALTER ... 60
Catalog Entry Types that Can Be Altered ... 60
ALTER Parameters: Summary ... 65
ALTER Parameters in Detail ... 66
BACKUP ... 72
BACKUP Parameters ... 73
BLDINDEX ... 77
BLDINDEX Parameters ... 77
CANCEL ... 79
DEFINE ALTERNATEINDEX ... 79
DEFINE ALTERNATEINDEX Parameters: Summary ... 82
DEFINE ALTERNATEINDEX Parameters ... 83
DEFINE CLUSTER ... 97
DEFINE CLUSTER Parameters: Summary ... 101
DEFINE CLUSTER Parameters ... 102
DEFINE MASTERCATALOG ... 120
DEFINE MASTERCATALOG Parameters: Summary ... 122
DEFINE MASTERCATALOG Parameters ... 123
DEFINE NONVSAM ... 129
DEFINE NONVSAM Parameters ... 130
DEFINE PATH ... 131
DEFINE PATH Parameters: Summary ... 131
DEFINE PATH Parameters ... 132
DEFINE SPACE ... 135
DEFINE SPACE Parameters ... 136
DEFINE USERCATALOG ... 139
DEFINE USERCATALOG Parameters: Summary ... 141
DEFINE USERCATALOG Parameters ... 142
DELETE ... 148
DELETE Parameters: Summary ... 150
DELETE Parameters ... 150
EXPORT ... 155
EXPORT Parameters ... 156
IMPORT ... 160
IMPORT Parameters ... 161
LISTCAT ... 167
LISTCAT Parameters ... 169
PRINT ... 171
Examples of PRINT Command Output ... 177
RECMAP ... 177
RECMAP Parameters ... 179
REPRO ... 181
REPRO Parameters ... 183
RESTORE ... 189

RESTORE Parameters: Summary....191
RESTORE Parameters.... 194
SNAP.... 199
SNAP Parameters....199
VERIFY.... 200
Modal Commands and their Formats....201
IF-THEN-ELSE.... 201
IF-THEN-ELSE Command Sequence.... 201
DO-END Command Sequence.... 204
SET....204
PARM.... 205

Chapter 3. Sample IDCAMS Command Job Streams....209
General Examples....209
Example 1: Define a System's Catalogs.... 209
Example 2: Define a VSE/VSAM User Catalog and a VSE/VSAM Data Space.... 210
Example 3: Define VSE/VSAM Files.... 212
Example 4: Define NonVSAM and VSE/VSAM Files....215
Example 5: Loading and Printing of Files.... 218
Example 6: Modifying and Printing the Contents of VSE/VSAM Files....221
Example 7: Modifying and Listing the Cataloged Attributes of a File.... 222
Example 8: Creating an Alternate Index and Its Path....223
Example 9: Defining a VSE/VSAM Data Space and Cluster on an FBA Volume....225
Example 10: Exporting a VSE/VSAM File....227
Example 11: Exporting an Alternate Index and Base Cluster....228
Example 12: Disconnecting a User Catalog from a Master Catalog.... 229
Example 13: Importing a Base Cluster and Alternate Index.... 229
Example 14: Importing an Entry-Sequenced File.... 230
Example 15: Connecting a User Catalog to the Master Catalog.... 231
Example 16: Using REPRO to Unload a User Catalog to Tape.... 232
Example 17: Using REPRO to Reload a User Catalog....232
Example 18: Deleting Entries in a User Catalog and the User Catalog Itself....233
Example 19: Defining an Entry Sequence Default Model.... 234
Example 20: Defining a Dynamic File.... 235
Example 21: Accessing a Dynamic File.... 235
Example 22: Defining a Partition-Independent File.... 236
Example 23: Syntax Checking a Command Stream.... 236
Example 24: Deleting Entries in the Master Catalog and the Master Catalog Itself.... 237
Example 25: IDCAMS PRINT That Allows Printing in Upper and Lower Case....239
Examples of Functions for BACKUP.... 240
Alternate Tape Support for Backup.... 240
Backup Cross-Reference Listings.... 240
Generic Names....242
Buffer Allocation for Backup....244
Backup Job Examples.... 244
Master Catalog.... 245
User Catalog.... 245
Backup Example 1....246
Backup Example 2....246
Backup Example 3....247
Backup Example 4....248
Backup Example 5....249
Backup Example 6....250
Backup Example 7....251
Backup Example 8....252
Examples of Functions for RESTORE.... 253
Generic Names....253

Restoration to a Volume of the Same Device Type ... 253
Restoration with File Modifications ... 254
Using RESTORE to Add Entries to a Catalog ... 255
Subsetting During Restoration ... 255
Reorganization of CAs ... 255
Considerations: Restoration from a Tape Resident Backup File ... 255
Considerations: Restoration from a Disk Resident Backup File ... 257
Buffer Allocation for Restoration ... 257
Restore Job Examples ... 258
Restore Example 1 ... 258
Restore Example 2 ... 259
Restore Example 3 ... 261
Restore Example 4 ... 261
Restore Example 5 ... 262
Restore Example 6 ... 263
Restore Example 7 ... 264
Restore Example 8 ... 265
Restore Example 9 ... 266
Restore Example 10 ... 266
Restore Example 11 ... 267
Restore Example 12 ... 268
Restore Example 13 ... 269

Appendix A. Interpreting LISTCAT Output ... 271
Entry Type Code Used with Field Group Descriptions ... 271
Description of Keyword Fields ... 271
The Allocation Group (D,I) ... 272
The Associations Group (G,R,C,D,I) ... 272
The Attributes Group (D,I,G,R) ... 273
The Data Space Group (V) ... 276
The History Group (C,D,G,I,R) ... 278
The NonVSAM Entry, Special Field For (A) ... 278
The Protection Group (C,D,G,I,R) ... 279
The Statistics Group (D,I) ... 279
The Volumes Group (D,I) ... 281
The Volume Entry, Special Fields For (V) ... 283
LISTCAT and IDCAMS Output Messages ... 284
LISTCAT Output Listing ... 284
LISTCAT VOLUME Output Listing ... 285
LISTCAT SPACE ALL Output Listing ... 288
LISTCAT ALL Output Listing ... 289
LISTCAT ALL Output Listing for Extra-Large Dataset ... 299
LISTCAT ALLOCATION Output Listing ... 304

Appendix B. Format of an Alternate-Index Record ... 311
System Header Information ... 311
Alternate Keys ... 311
Alternate-Index Pointers ... 311

Notices ... 313
Programming Interface Information ... 314
Trademarks ... 314
Terms and Conditions for Product Documentation ... 314

Accessibility ... 317
Using Assistive Technologies ... 317
Documentation Format ... 317

Glossary **319**

Index **331**

Figures

1. Defining an Extra-Large Dataset 28

2. Job Stream Example for Defining a KSDS File with Extended Addressing 29

3. Example: Defining Work Files on Virtual Disk 38

4. Data Portability by Moving Volumes or Files 44

5. JCL for BACKUP to Tape 54

6. JCL for RESTORE to Tape 55

7. Output Example of PRINT DUMP Command 177

8. Format of Blocked and Unblocked Records 184

9. Alternate Tape Support for Backup 240

10. Backup Volume Cross-Reference Listing (Tape Resident Backup File) 240

11. Backup Object Cross-Reference Listing (Tape Resident Backup File) 241

12. Backup Extent Cross-Reference Listing (Disk Resident Backup File) 241

13. Backup Object Cross-Reference Listing (Disk Resident Backup File) 241

14. Backup Extent List (Disk Resident Backup File) 242

15. Generic Backup 243

16. Exclusion from Backup 243

17. Automatic Backup of Alternate Indexes and Paths 244

18. Exclusion from Automatic Backup 244

19. Generic Restoration 253

20. Restoration of an Alternate Index Excluding the Base Cluster 253

21. Alternate Tape Support for Restoration 258

22. Example of Messages That Follow the Entries of Output Listings 284

23. Example of LISTCAT Output When No Parameters are Specified 285

24. Example of LISTCAT VOLUME Output..... 285

25. Example of LISTCAT SPACE ALL Output.....288

26. Example of LISTCAT ALL Output..... 290

27. Example of LISTCAT ALL Output for Extra-Large Dataset..... 300

28. Example of LISTCAT ALLOCATION Output.....304

Tables

1. Functions of IDCAMS Commands 1

2. Password Requirements for IDCAMS Commands 9

3. Minimum Space Allocation for Catalogs 16

4. Example: Empty Versus Nonempty Output File 43

5. ALTER Parameters and Their Objects 61

6. When to Specify DLBL and CATALOG Parameter 77

7. Minimum CA and Maximum CA for FBA Devices 103

8. Beginning and Ending Blocks for FBA Devices 103

9. Minimum CI Sizes Depending on Key Length 110

10. Listing Contents by Parameters 167

11. Abbreviations for the TYPE Parameter 181

12. Files and Keywords with FROM-ADDRESS, -KEY, -NUMBER 186

13. Files and Keywords with TO-ADDRESS, -KEY, -NUMBER, COUNT 188

About This Publication

This publication explains in detail the functions that are available with the *IBM VSE/Virtual Storage Access Method (VSE/VSAM)*. It describes the *IDCAMS commands*, and it includes information on *using* the commands.

This publication is intended for the VSE/VSAM administrator. It explains in detail the functions of the IDCAMS commands and how to use them. IDCAMS is a utility program within VSE/VSAM for defining files and catalogs, and for defining the address space needed for VSE/VSAM objects.

For guidance information and for details about VSE/VSAM macros, refer to the VSE/VSAM User's Guide and Application Programming.

For information on VSE/VSAM messages (IDCAMS and macros), refer to z/VSE Messages and Codes Volume 2 .

Who Should Use This Publication

This publication is intended for the VSE/VSAM administrator.

How to Use This Publication

- Chapter 1, “The IDCAMS Utility Program,” on page 1 provides an overview of IDCAMS function and use including general language considerations. It gives examples of using IDCAMS commands and of z/VSE job control to be used with them.
- Chapter 2, “The IDCAMS Commands,” on page 59 shows the format for each functional and modal command and the meaning of each parameter.
- Chapter 3, “Sample IDCAMS Command Job Streams,” on page 209 shows sample job streams for IDCAMS commands including BACKUP and RESTORE.
- Appendix A, “Interpreting LISTCAT Output,” on page 271 shows and explains the output from the LISTCAT command of the IDCAMS utility.
- Appendix B, “Format of an Alternate-Index Record,” on page 311 shows the format of an alternate-index record for use in defining the RECORDSIZE parameter in the DEFINE ALTERNATEINDEX command.

Where to Find More Information

- VSE/VSAM User's Guide and Application Programming
  Contains guidance information for using the functions that are available with IBM VSE/VSAM. It describes the major facilities of the program and how to use them efficiently. The publication also explains concepts of VSE/VSAM, and includes information on planning, and using diagnosis tools. This publication also includes the VSE/VSAM macros and explains how to use them (to access data from an assembler program).
- z/VSE Messages and Codes Volume 2

  Contains messages and return codes from VSE/VSAM.
  Refer to the following topics:

  – For return codes from VSE/VSAM macros refer to "VSAM Codes"
  – For messages issued by VSE/VSAM refer to the following sections:
    - "4-Prefix" messages, especially the sections for 42xx and 4Axx
    - "IDC-Prefix"
    - "IDCAMS Codes" for VSE/VSAM catalog management return and reason codes

– For messages issued by the VSE/VSAM Catalog Check Service Aid (IKQVCHK), refer to the "IKQ-Prefix" section.

z/VSE IBM Documentation

IBM Documentation is the new home for IBM's technical information. The z/VSE IBM Documentation can be found here:

https://www.ibm.com/docs/en/zvse/6.2

You can also find VSE user examples (in zipped format) at

https://public.dhe.ibm.com/eserver/zseries/zos/vse/pdf3/zVSE_Samples.pdf

Summary of Changes

Note:

- z/VSE 6.1 requires initial installation.
- Fast Service Upgrade (FSU) to z/VSE 6.1 is not possible.

This publication has been updated to reflect terminology, maintenance, and editorial changes.

What is New With z/VSE 6.1?

For a complete overview of the functions that are new with z/VSE 6.1, refer to the z/VSE Release Guide.

Chapter 1. The IDCAMS Utility Program

IDCAMS is a utility program that is part of VSE/VSAM. It serves to create and maintain files.

You can invoke IDCAMS functions in a job step that contains IDCAMS commands and their parameters, and that specifies:

```
// EXEC IDCAMS,SIZE=AUTO
```

For more information, refer to the VSE/VSAM User's Guide and Application Programming under "// EXEC Statement".

You can also call the IDCAMS program from within another program and pass a command and its parameters to the IDCAMS program. For details, refer to the VSE/VSAM User's Guide and Application Programming under "Invoking IDCAMS from a Program".

Resource Access Control Facility interface for IDCAMS commands has been implemented to integrate VSAM into the overall z/VSE BSM concept. Access control on the level of IDCAMS has been added. Refer to the VSE/VSAM User's Guide and Application Programming under "IDCAMS Commands Security".

Functions of IDCAMS

Table 1 on page 1 lists the functions that IDCAMS commands can perform.

Table 1. Functions of IDCAMS Commands

Function	Command
Add a password to a new VSE/VSAM file or catalog	DEFINE
Add a password to an existing VSE/VSAM file or catalog	ALTER
Attach a user catalog to the master catalog	DEFINE, IMPORT
Backup all VSE/VSAM objects.	BACKUP
Build an alternate index.	BLDINDEX
Cancel a job or job step.	CANCEL
Catalog a VSE/VSAM file.	DEFINE
Catalog a nonVSAM file.	DEFINE
Change a file's description in the catalog.	ALTER
Change the device type of the volume on which the catalog resides	REPRO
Change a password.	ALTER
Connect a user catalog to a master catalog	IMPORT
Convert a SAM or ISAM file to VSE/VSAM format.	REPRO
Convert a VSE/VSAM file to sequential format.	REPRO
Copy a file.	REPRO
Create an alternate index.	DEFINE, BLDINDEX
Create a backup copy of a catalog.	REPRO

Table 1. Functions of IDCAMS Commands (continued)

Function	**Command**
Create a backup copy of a VSE/VSAM file.	REPRO, EXPORT
Create a catalog.	DEFINE
Create a catalog entry for a nonVSAM file.	DEFINE
Create a data space.	DEFINE
Create a nonVSAM file.	DEFINE
Create a path.	DEFINE
Create a VSE/VSAM file.	DEFINE
Define See "Create"	
Delete an alternate index.	DELETE
Delete a catalog.	DELETE
Delete a data space.	DELETE
Delete a nonVSAM file.	DELETE
Delete a password.	ALTER
Delete a path.	DELETE
Delete a VSE/VSAM file.	DELETE
Disconnect a user catalog from the master catalog	EXPORT
List a password.	LISTCAT
List a file.	PRINT
List a file's catalog entry.	LISTCAT
List contents of the catalog.	LISTCAT
Load records into a file.	REPRO
Load a catalog from an unloaded copy.	REPRO
Modify a file's description in the catalog.	ALTER
Move a catalog to another system.	EXPORT, IMPORT
Move a file to another system.	EXPORT, IMPORT
Print a file	PRINT
Recover from loss of data due to improper closing of a file.	VERIFY
Recreate a VSE/VSAM file from a back up copy.	IMPORT
Map a VSAM cluster to a relational structure, and later maintain the associated map or view.	RECMAP
Release a user catalog from the master catalog.	EXPORT
Reload a catalog from a sequential file.	REPRO
Reorganize a file.	REPRO, or
Rename a file.	ALTER

Table 1. Functions of IDCAMS Commands (continued)	
Function	**Command**
Restore all VSE/VSAM objects.	RESTORE
Snap (copy) one or more ESS volumes.	SNAP
Terminate the relation that was established when creating the snapshot.	SNAP
Uncatalog a file.	DELETE
Unload a file.	REPRO
Unload a catalog to a sequential file.	REPRO
Verify end-of-file.	VERIFY
To change program execution, you can use the following modal commands:	
Change flow of control.	IF–THEN–ELSE
Control command execution.	IF–THEN–ELSE
Set/reset condition codes.	SET
Specify diagnosis tools.	PARM
Specify printed output.	PARM
Specify processing options.	PARM
Specify syntax checking.	PARM

Types of IDCAMS Commands

The IDCAMS commands are of two types:

- **Functional** commands. Use these commands to specify what you want IDCAMS to do. For example, add a password, create a file, or list the contents of a catalog.
- **Modal** commands. Use these commands to specify options and conditional execution of the functional commands.

Functional Commands

The functional commands are:

ALTER
to alter existing catalog entries.

BACKUP
to write a file or index to tape or disk.

BLDINDEX
to build alternate indexes for existing VSE/VSAM files.

CANCEL
to cancel a job or job step.

DEFINE
to create catalogs and catalog entries for files, alternate indexes, paths, and VSE/VSAM data spaces.

DELETE
to delete catalog entries.

EXPORT
to create a copy of a file for backup or to make a file or user catalog portable so that it can be used in another system.

IMPORT
to read a backup copy of a file or to make a file or user catalog previously exported available in another system.

LISTCAT
to list catalog entries.

PRINT
to print VSE/VSAM and nonVSAM files.

RECMAP
to map a VSE/VSAM cluster to a relational structure and later maintain the associated map or view.

REPRO
to copy files, to convert sequential files on tape or disk and indexed-sequential files to VSE/VSAM format, to convert VSE/VSAM and indexed-sequential files to sequential format, to create backup copies of VSE/VSAM catalogs, and to reload a VSE/VSAM catalog from a backup copy.

RESTORE
to write a file or index from tape or disk to disk. To print the list of objects backed up.

SNAP
to create a snapshot of entire ESS volumes on which the catalog and all its data sets reside. To terminate the relation that was established when creating the snapshot.

VERIFY
to cause a catalog to correctly reflect the end of a file. You should use this command after an error occurred which prevented normal closing of the file. This may have caused the catalog to be incorrect.

On nonVSAM files, you can use the functional commands ALTER, DEFINE, DELETE, LISTCAT, PRINT, and REPRO.

Modal Commands

The modal commands are:

IF
tests a condition code and executes according to the results of the test.

IF is followed by THEN and ELSE clauses which specify alternative actions.

DO and END
specify the beginning and ending of a functional command sequence, normally within a THEN or ELSE clause.

SET
changes condition codes.

PARM
specifies diagnosis tools, syntax checking and printed output options, and changes input record margins.

For more information on these commands, refer to “Modal Commands and their Formats” on page 201.

Coding of IDCAMS Commands

Understanding Syntax Diagrams

This section describes how to read the syntax diagrams in this publication.

To read a syntax diagram follow the path of the line. Read from left to right and top to bottom.

- The ▶▶── symbol indicates the beginning of a syntax diagram.

- The ⟶ symbol, at the end of a line, indicates that the syntax diagram continues on the next line.
- The ⟶ symbol, at the beginning of a line, indicates that a syntax diagram continues from the previous line.
- The ⟶◂ symbol indicates the end of a syntax diagram.

Syntax items (for example, a keyword or variable) may be:

- Directly on the line (required)
- Above the line (default)
- Below the line (optional)

Uppercase Letters

Uppercase letters denote the shortest possible abbreviation. If an item appears entirely in uppercase letters, it can not be abbreviated.

You can type the item in uppercase letters, lowercase letters, or any combination. For example:

In this example, you can enter KEYWO, KEYWOR, or KEYWORD in any combination of uppercase and lowercase letters.

Symbols

You **must** code these symbols exactly as they appear in the syntax diagram

*****
: Asterisk

:
: Colon

,
: Comma

=
: Equal Sign

-
: Hyphen

//
: Double slash

()
: Parenthesis

.
: Period

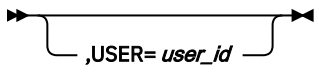
+
: Add

For example:

```
* $$ LST
```

Variables

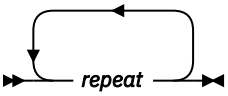
Highlighted lowercase letters denote variable information that you must substitute with specific information. For example:



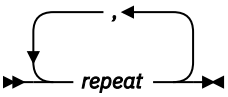
Here you must code USER= as shown and supply an ID for user_id. You may, of course, enter USER in lowercase, but you must not change it otherwise.

Repetition

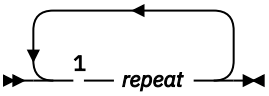
An arrow returning to the left means that the item can be repeated.



A character within the arrow means you must separate repeated items with that character.



A footnote (1) by the arrow references a limit that tells how many times the item can be repeated.

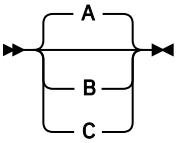


Notes:

[1] Specify *repeat* up to 5 times.

Defaults

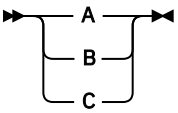
Defaults are above the line. The system uses the default unless you override it. You can override the default by coding an option from the stack below the line. For example:



In this example, A is the default. You can override A by choosing B or C.

Required Choices

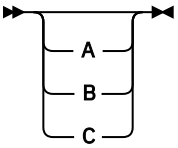
When two or more items are in a stack and one of them is on the line, you **must** specify one item. For example:



Here you must enter either A or B or C.

Optional Choice

When an item is below the line, the item is optional. Only one item **may** be chosen. For example:



Here you may enter either A or B or C, or you may omit the field.

Required Blank Space

A required blank space is indicated as such in the notation. For example:

```
* $$ EOJ
```

This indicates that at least one blank is required before and after the characters $$.

Syntax of IDCAMS Commands

All IDCAMS commands have this general structure:

```
VERB   parameter(s) terminator
```

VERB specifies the type of service requested. Parameter(s) further describe the service requested. Terminator indicates the end of the command. For examples, see Chapter 3, “Sample IDCAMS Command Job Streams,” on page 209.

You can abbreviate many of the verbs and parameters. Permitted abbreviations are listed in the description for every command or parameter.

Verbs

Code the verbs beginning at or to the right of the left margin. The default columns are column 2 through 72. You can specify different columns, if you want, with the PARM command.

Separate every verb from its parameter(s) either with a blank or comment.

Comments

You can add comments to a command anywhere a blank can appear. Comments are strings of characters preceded by a /* and followed by an */. They can contain any characters you want, except an */. You continue comments to the next line by using a hyphen (-) or plus (+) sign as the last nonblank character before or at the right margin.

Parameters

One or more parameters follow the verb. They can be *positional parameters* or *keyword parameters*.

Positional Parameters

Such parameters follow the verb in a prescribed sequence. An example of a positional parameter in this publication is the *entryname* parameter in the ALTER, DELETE, and EXPORT commands. It must always be the first parameter following the verb. If a positional parameter consists of a list of names, enclose the list in parentheses. A single item, however, does not require parentheses.

Keyword Parameters

Such parameters are specific names that have a particular meaning to IDCAMS. You can include these parameters in any order following the positional parameter (if present), or the verb. Keyword parameters are shown in this publication in uppercase type; for example: ERASE.

Some keyword parameters also require values or names. In that case, you enclose the value or name in parentheses following the parameter; for example: DEVICETYPE(3380).

If the value you specify contains commas, semicolons, blanks, parentheses, or slashes, you enclose the entire value in single quotation marks.

Subparameters of a Keyword Parameter

A keyword parameter may also have a set of subparameters; for example:

```
ENV(PDEV(3390)  RECFM(VARUNB)  BLKSZ(5164))
```

The following are subparameters of the ENV (ENVIRONMENT) keyword of the preceding example (every parameter is abbreviated so that the example fits on a single line):

PDEV (PRIMEDATADEVICE)
RECFM (RECORDFORMAT)
BLKSZ (BLOCKSIZE)

Set of Parameters and Subparameters

You can code a list of similar items (for example, volume numbers) in a parameter or subparameter. Separate the parameters and subparameters from one another by one or more separators (commas, blanks, or comments). The only exception is that parameters immediately preceding or following a subparameter set enclosed in parentheses do not need to be separated from the opening or closing parenthesis.

Password

In some cases it is necessary to specify a password following the *name* of a catalog entry or of a job control statement. You do this by coding:

(1) The name
(2) A slash
(3) The password

The name/password combination must be preceded and followed by separators; for example:

```
DELETE  PAYROLL/CTLGPAY
```

Continuing Command Statements

When you continue a statement on several coding lines (coding line is sometimes referred to as a record), you must use a hyphen (-) or plus (+) sign to indicate the continuation of the command. (The coding examples in this publication generally use a separate line for the verb and every one of its parameters. This makes it easier to read the command.) The plus sign differs from the hyphen because not only does it indicate continuation of the command but it also indicates continuation of a value within the command.

Blank coding lines or coding lines ending with complete comments (that is, end of comment: */) must also end with a continuation mark when they appear in the middle of a command and when they appear preceding or following the THEN and ELSE clauses of an IF command.

Records ending with partial comments must always end with a continuation mark.

Also, only blank characters may appear between a continuation mark and the end of the coding line.

Common Continuation Errors

The continuation rules must be used carefully when modal commands, comments, or blank records appear in the input stream. You must be careful when continuing a modal command so that you do not inadvertently specify a null clause. For information on null clauses, see “Modal Commands and their Formats” on page 201.

The following examples show common continuation errors. For examples of correct coding, refer to VSE/VSAM User's Guide and Application Programming.

```
IF LASTCC = 0 -
THEN
LISTCAT
```

A continuation mark (hyphen) is missing after the THEN keyword. A null clause is assumed after the THEN keyword, and the LISTCAT command is unconditionally executed.

```
IF LASTCC = 0 -
THEN          -
REPRO INFILE(F1)OUTFILE(F2)
/*ALTERNATE PATH*/
ELSE          -
PRINT ...
```

Because no continuation mark (hyphen) follows the comment, a null ELSE clause is assumed. The ELSE keyword will not match up with the THEN keyword, an error message will be issued, and the PRINT command ignored. Note the correct use of the continuation marks on the other records.

```
IF LASTCC = 0 -
THEN          -
REPRO ...
ELSE          -

PRINT ...
```

Because a blank line with no continuation mark (hyphen) follows the ELSE keyword, the ELSE becomes null and the PRINT command is unconditionally executed.

```
PARM TEST ( - /*COMMAND*/
TRACE)
```

The PARM command will not be continued to the second record because characters other than blanks appear between the continuation mark (hyphen) and the end of the record.

```
PARM TEST ( TRA+
/*FIELD CONTINUATION*/
CE)
```

The end of the PARM command is found after the second record because no continuation was indicated after the comment. The command is rejected.

Terminator

The terminator indicates the end of the command. The terminator can be a semicolon or simply the absence of a continuation mark. If you use the semicolon as the terminator, the semicolon cannot be enclosed in quotation marks or embedded in a comment. Everything to the right of the semicolon is ignored. If there is information to the right of the semicolon that is continued to another record, all of the information including the continued information is ignored. For example, if you code:

```
PARM TEST (TRACE); PARM          -
  GRAPHICS (CHAIN(TN))/*COMMENT*/-
PRINT ...
REPRO ...
```

the characters following the semicolon terminator are ignored. The continuation marks (hyphens) at the end of the first and second records cause the PRINT command to be ignored also. The first PARM command and the REPRO command are the only commands that are recognized.

Password Requirements for IDCAMS Commands

The administrator specifies the passwords, and the system checks them when any user wants access to the files.

Table 2 on page 9 summarizes which passwords may be used when using the IDCAMS commands. For explanations to (n) in the figure, see "Notes" after the figure.

Where several passwords are indicated, any of them may be used. For example: to alter an entry in a password-protected catalog, you must supply the master password of either the catalog or cluster/AIX.

Table 2. Password Requirements for IDCAMS Commands

	Type of Catalog Entry				CLUSTER/AIX			
	MASTER PW	**UPDATE PW**	**CTLPW**	**READ PW[6]**	**MASTER PW**	**UPDATE PW**	**CTLPW**	**READ PW[6]**
ALTER[1]	x				x			
BLDINDEX								
Input File					x	x	x	x
Output File					x	x	x	
Sort Work File	x	x	x					

Table 2. Password Requirements for IDCAMS Commands (continued)

	Type of Catalog Entry				CLUSTER/AIX			
	MASTER PW	**UPDATE PW**	**CTLPW**	**READ PW[6]**	**MASTER PW**	**UPDATE PW**	**CTLPW**	**READ PW[6]**
DEFINE								
AIX® or PATH[2]	x	x	x		x			
Cluster	x	x	x					
NonVSAM	x	x	x					
Space	x	x	x					
User Catalog	x	x	x					
DELETE								
Catalog or Path	x							
Space empty	x	x	x					
Space non empty	x							
Cluster, AIX[1]	x				x			
EXPORT								
User Catalog	x	x	x					
Cluster, AIX[4]	x				x			
IMPORT[5]	x	x	x		x	x	x	
LISTCAT								
- With Passwords	x							
- Without Passwords		x	x	x				
Entries:								
- With Passwords[1]	x				x			
- Without Passwords[1]		x	x	x		x	x	x
PRINT								
Catalog[3]	x							
Cluster					x	x	x	x
Component					x	x	x	x
REPRO								
Input File								x
Output File					x	x	x	
Catalog	x				x	x	x	
VERIFY								
File (Cluster, AIX, Component)					x	x		

Notes on Table 2 on page 9:

- (1) Either catalog or cluster password may be specified.
- (2) Both the catalog and the cluster password are checked if both are password-protected.
- (3) The master password is required only if passwords are to be printed; otherwise the read password may be used.
- (4) If any password-protected paths are defined over the cluster or alternate index being exported, you must supply a master password of the catalog to export the protection attribute of all paths.
- (5) Either the catalog or cluster password can be specified. If you are importing into a predefined empty file with passwords different from the old ones of the file being imported, you must specify the file's update or higher-level password.
- (6) The READPW is inadequate if the DLBL statement describing the file specifies one of the following:

```
DISP=NEW
DISP=(,DELETE)
DISP=(,DATE)
```

In these cases, you must provide the UPDATEPW (or higher-level password).

For more information on password specification, refer to the VSE/VSAM User's Guide and Application Programming to the "Data Protection" section .

Job Control

IDCAMS DEFINE commands and job control statements are used to set up all catalogs, VSE/VSAM data spaces, VSE/VSAM files (clusters), alternate indexes, paths, and nonVSAM files.

A VSE/VSAM file exists once it has been defined in a catalog. Before you can define a file, you must have a catalog in which to define it. Generally you must also have a VSE/VSAM data space on a disk volume in which to suballocate space for the file (as opposed to defining a catalog or defining a file in its own unique data space). Once a VSE/VSAM file has been defined, records can be loaded into it.

When you invoke IDCAMS using job control, you must specify an EXEC job control command or statement:

```
// EXEC  IDCAMS,SIZE=AUTO
```

You must specify the SIZE parameter as shown. Otherwise, IDCAMS will terminate your job immediately.

IDCAMS Input and Output Files

When a VSE/VSAM file is to be opened for access by the IDCAMS command PRINT or REPRO, you must supply the following job control information:

```
// DLBL filename,'file-ID',,VSAM
```

The *filename* identifies the file and matches the dname of an INFILE or OUTFILE parameter in the command. The *file-ID* is identical to the name of the file (entryname, catname, or newname parameter) specified in the commands of IDCAMS and listed in the VSE/VSAM catalog.

Optionally, the BUFSP, BUFND, and BUFNI parameters of the // DLBL statement can be specified to override:

- The cataloged BUFFERSPACE parameter (only if the BUFSP parameter is equal to, or greater than the cataloged BUFFERSPACE parameter).
- The buffer space value (in the ACB macro) to which IDCAMS defaults.

Note: Because EXPORT and IMPORT always default to an optimum buffer space value (through the ACB macro parameter), VSE/VSAM ignores any BUFSP specification made by you in the // DLBL statement associated with these commands. All other commands use the VSE/VSAM default values in their ACB macros.

For more information on buffer space options and defaults, refer to the VSE/VSAM User's Guide and Application Programming under "The ACB Macro".

You must use SYSLST as the output file for listing. Print lines are 121 bytes in length. The first byte is the ASA control character. The parameters of this file are listed below. Only the number of lines per page can be changed.

- Record format: Fixed, unblocked with ASA carriage control
- Logical record length: 121
- Block size: 121
- Printed lines per page: 56. Use the job control SET LINECT command if you want to specify the value (between 30 and 99) for SYSLST printed lines per page.

For a description of the job control statements to be used for VSE/VSAM data spaces and files, refer to "Defining Objects in a Catalog" on page 15.

Tape Considerations for IDCAMS

To process and create files on magnetic tape, use commands of the IDCAMS utility program. The following explanations show what you need to consider and how to proceed.

VSE/VSAM Support for Tape Labels

- VSE/VSAM does *not* support:
  - American National Standard (ASCII) labels (DTFMT macro, ASCII=YES).
  - Nonstandard tape labels (DTFMT macro, FILABL=NSTD).
- User standard labels (DTFMT macro, LABADDR=xxxxx) are bypassed on input, and are *not* supported on output.
- The maximum block size is 32,767 (32KB - 1) except for BACKUP/RESTORE, where the maximum block size is 65,535 (64KB -1).

IDCAMS Commands that Relate to Tape Processing

For tape input and output, use the following IDCAMS commands:

- EXPORT for tape output.
- IMPORT for tape input.
- REPRO for tape input or output.

Each of these commands has the ENVIRONMENT parameter. This parameter, in turn, has **subparameters** where you specify your requirements for tape input/output. The subparameters have the foll owing purpose:

NOLABEL and STDLABEL

To process an existing tape file, or to create a new tape file without tape labels (NOLABEL) or with EBCDIC standard labels (STDLABEL).

NOREWIND, REWIND, and UNLOAD

To control tape positioning for OPEN, CLOSE, and end-of-volume (EOV) processing, where:

- NOREWIND indicates that rewind is never performed for OPEN, CLOSE, and EOV processing. You must specify NOREWIND if you wish to create or process multifile tape volumes.

  It is your responsibility to ensure that a tape is properly positioned before processing it with IDCAMS.
- REWIND indicates that tapes are rewound to the load point, but are not unloaded on the conditions: OPEN, CLOSE, and EOV.

  You should specify REWIND (or UNLOAD) whenever you wish to create or process single file tape volumes.
- UNLOAD indicates that tapes are rewound on an OPEN, and rewound and unloaded on the conditions: CLOSE and EOV.

  You should specify UNLOAD (or REWIND) whenever you wish to create or process single file tape volumes. Specify UNLOAD if you desire demounting when file creation or processing is complete.

If *VSE/Access Control-Logging and Reporting* (IBM VSE/ACLR) is installed, and if you wish to process unlabeled tapes, you must specify REWIND or UNLOAD.

Tape Assignments for IDCAMS

You must assign magnetic tape input files to SYS004 and magnetic tape output files to SYS005.

Multivolume File Considerations for IDCAMS

If you process or create a file that requires several tape volumes, specify (or default to) the STDLABEL option. This is necessary because IDCAMS cannot distinguish between the end-of-file (EOF) and the end-of-volume (EOV) on a tape that has no label; IDCAMS always assumes EOF.

When processing multivolume files, it is recommended to specify a volume-sequence number of 1 (one) in the // TLBL statement. This ensures that the first volume is mounted.

Multifile Volume Considerations for IDCAMS

Processing Input Files (Tape has Standard Labels)

To Process the First File on a Tape:

1. Before processing, ensure that the tape is at the load point. Use the statement: // MTC REW,SYS004
2. Specify STDLABEL and NOREWIND (so that the succeeding file can be processed next).

To Process any Other File on a Tape:

- If the previous statement (or program) has processed the previous file, but has not rewound the tape, specify STDLABEL and NOREWIND. This maintains positioning for OPEN, and positions in front of the following file (if any) at CLOSE.
- If the tape is not positioned at the beginning of the file:
  1. Rewind the tape. Use the statement: // MTC REW,SYS004
  2. Use either of the following statements:
     - To identify the required file, use the: *file-sequence-number* parameter of the // TLBL statement.
     - To position to the beginning of the desired file, use the statement:

       // MTC FSF,SYS004,nn
       where: nn must equal three times the number of files to be bypassed
       (for example, to skip past 2 files, specify 6).
  3. Specify STDLABEL and NOREWIND (so that the succeeding file can be processed next).

Creating Output Files (Tape has Standard Labels)

To Create the First File on a Tape:

1. Ensure that a standard volume label (VOL1) has previously been written as the first record on the selected tape volume.
2. Ensure that the tape is at the load point. Use the statement: // MTC REW,SYS005
3. Specify STDLABEL and NOREWIND. This positions the tape for succeeding statements.
4. In the // TLBL statement of the file, either omit the file-sequence-number, or specify 1.

To Create any Other File on a Tape:

- If the previous statement (or program) has processed the previous file, but has not rewound the tape, specify STDLABEL and NOREWIND. This maintains positioning for OPEN, and positions the tape for a following file (if any) at CLOSE.
- If the tape is positioned incorrectly:
  1. Rewind the tape. Use the statement: // MTC REW,SYS005

     This rewinds the tape to the load point.
     It positions the tape just beyond the tape mark that follows the
     EOF1 trailer record of the previous file.
  2. Correctly position the tape.

Use the statement: // MTC FSF,SYS005,nn
where: nn is equal to three times the number of files
on the tape preceding the file to be created.

3. Specify STDLABEL and NOREWIND.

   This maintains positioning for OPEN, and positions the tape
   for a following file (if any) at CLOSE.

In all cases, you can omit the file-sequence-number parameter of the // TLBL statement.

Processing/Creating Input/Output Files (Tape is Unlabeled)

The following explanations apply to both, *processing* and *creating* files on a tape; the tapes are *unlabeled*.

To Process/Create the First File on a Tape:

1. Ensure that the tape is at the load point.
   - For input, use the statement: // MTC REW,SYS004
   - For output, use the statement: // MTC REW,SYS005
2. Specify NOLABEL and NOREWIND. This allows the succeeding file to be processed or created next.

To Process/Create any Other File on a Tape:

- If the previous statement (or program) has processed or created the previous file, but has not rewound the tape, specify NOLABEL and NOREWIND. This maintains positioning for OPEN, and positions in front of the following file (if any) at CLOSE.
- If the tape is not positioned at the beginning of the file:
  1. Rewind the tape.

     Use the statement: // MTC REW,SYSxxx
     where:

     xxx is 004 for input
     xxx is 005 for output

  2. Position to the beginning of the desired file. Use the statement: // MTC FSF,SYSxxx,nn where:

     xxx is 004 for input
     xxx is 005 for output

     For files created with IDCAMS, nn must equal the number of files to be bypassed (IDCAMS uses the DTFMT macro TPMARK=NO option). For every file to be bypassed, but created with the DTFMT macro TPMARK=YES specification, add two to nn.

Device Dependencies for IDCAMS

Large DASD

Support is provided for DASD with capacities exceeding 65535 (64K) tracks. For such devices, the level of granularity for space allocation is cylinders rather than tracks. VSE/VSAM supports up to 65520 cylinders per ECKD device.

Support is divided into BIG- & FAT-DASD:

- BIG-DASD=64K tracks -10017 cylinders
- FAT-DASD=64K tracks -65520 cylinders

SCSI Disk Devices

z/VSE® 3.1 is designed to allow IBM eServer™ zSeries servers to attach industry-standard Small Computer System Interface (SCSI) disk devices via zSeries Fibre Channel Protocol (FCP) channels.

User-written programs use VSE's existing Fixed Block Architecture (FBA) support (512 byte blocks) to access SCSI disks. User programs cannot use SCSI commands directly.

z/VSE 3.1 is designed to support SCSI disk volume sizes from 8 MB to 24 GB. Because z/VSE itself uses the first 4 MB for internal purposes, the available user space is equal to the defined size of the disk minus 4 MB. z/VSE 3.1 limits VSE/VSAM to the first 16 GB of any SCSI volume. Such devices are treated by z/VSE as standard FBA devices. Unless otherwise stated, information in this publication concerning FBA devices also applies to SCSI devices.

Defining Objects in a Catalog

VSE/VSAM uses catalogs as central information points for files and the disk volumes on which they are stored. You can define a VSE/VSAM object in a VSE/VSAM catalog by using the IDCAMS DEFINE command. Additionally, you can use the DEFINE command to define nonVSAM objects in a VSE/VSAM catalog.

When you enter the DEFINE command, IDCAMS builds a catalog entry for the object. The VSE/VSAM objects you can define are:

- **Master catalog**, which is the primary VSE/VSAM catalog. It contains information about VSE/VSAM objects in the system. You must create a VSE/VSAM master catalog before you can define any other object.
- **User catalog**, which contains information about VSE/VSAM objects that reside on the volumes on which the user catalog owns space. You can create a user catalog after the VSE/VSAM master catalog is defined. A pointer to the user catalog is put in the master catalog.
- **Data space**, which is disk space used for a catalog, for clusters, and for alternate indexes. VSE/VSAM controls the space allocation within each data space.
- **Cluster** or **VSE/VSAM file**, which is a collection of user-data records. There are three types of cluster data organization:
  - Entry-sequenced, or sequential, in which data records are read or written sequentially.
  - Key-sequenced, or indexed, in which a data record is read or written based on its key value. A key is a field, shorter than and within the record, that identifies the record.
  - Relative-record, or direct, in which a data record is read or written based on its relative record number— its displacement, in records, from the beginning of the cluster.
- **Alternate index**, which allows you to read and write data records in an entry-sequenced or key-sequenced cluster (called the base cluster) based on an alternate key.
- **Path**, which is a file name for the combination of an alternate index and its base cluster or an alias for a VSE/VSAM file. A path entry can be password-protected.
- **NonVSAM file**, which is a file that is not in VSE/VSAM format, but can be cataloged in a VSE/VSAM catalog.

When you define an object, you specify attributes to be associated with it. The attributes include, for example, any passwords required to use data and how space is to be allocated. After the object is defined, it can be processed with other IDCAMS commands and with the user's VSE/VSAM program. After a cluster is defined, for example, you can load data records into it by using the REPRO command.

VSE/VSAM clusters, alternate indexes, and catalogs are stored in data spaces. Usually you define a data space first, then define the files. To define a VSE/VSAM file as the only one in its data space, specify the parameter UNIQUE when you define the file.

Defining an object normally does not require that a volume be mounted. This is because IDCAMS can determine the availability of space in data spaces by examining the volume information in the catalog. However, a volume must be mounted whenever a data space, unique file, or catalog is being defined, deleted, or altered.

If the volume is not mounted, VSE/VSAM issues a mount message. For more information, see the VSE/VSAM User's Guide and Application Programming under "Volume Mounting Needs".

You can use the entry of an already-defined VSE/VSAM object (that is an already-defined alternate-index, catalog, cluster, or path) as a model for the definition of another object of the same type. When one entry is used as a model for another, its attributes are copied as the new entry is defined.

Defining a Catalog

The DEFINE command is used to define the master catalog or user catalogs. User catalogs are pointed to by the master catalog.

The data space that contains the catalog is allocated when the catalog is defined. The data space may be reserved for the catalog's exclusive use or it may contain other VSE/VSAM files.

Catalog Space Estimates and Worksheet

The control intervals (CIs) in a catalog are 512 bytes long, and each CI contains one record.

A catalog requires at least *three* control areas (CAs). The CA size depends on the device type.

For a discussion of minimum CAs and maximum CAs, see the VSE/VSAM User's Guide and Application Programming under "Control Area (CA) Size".

To estimate the disk space required for a catalog, refer to Table 3 on page 16.

Table 3. Minimum Space Allocation for Catalogs

Device Type	Data Allocation	Index Allocation	Total
CKD/ECKD (Small DASD)	4 tracks	2 tracks	6 tracks
ECKD (Large DASD: BIG & FAT)	60 tracks = 4 cylinders	15 tracks = 1 cylinder	5 cylinders
Generic FBA	256 blocks	128 blocks	384 blocks
FBA-SCSI	2048 blocks	1024 blocks	3072 blocks

The Master Catalog

The first job you run after you have installed VSE/VSAM and IDCAMS in your system is the one that creates your master catalog. The volume on which the master catalog is defined must be mounted whenever VSE/VSAM is used; it is always on a logical unit named SYSCAT.

You can have several master catalogs at your installation; however, only one can be connected to the system at a time. Also, you cannot define these extra master catalogs while the first one is connected. The master catalog is connected to the system during system start up by the DEF SYSCAT=cuu command.

How Data Space is Assigned to a Catalog

When a master or user catalog is defined, the catalog is the first VSE/VSAM object contained on the volume. The data space that contains the catalog is built when the catalog is defined.

Three important points about a catalog and its data space:

1. VSE/VSAM allocates a specific amount of data space to a catalog (the catalog "owns" this space).
2. All or a portion of that specific amount of data space can be made available (suballocated) for the catalog itself. If only a portion of the total amount of data space is suballocated to the catalog, the remaining data space (still owned by the catalog) is available for future expansion of the existing catalog or for other VSE/VSAM objects.
3. Several catalogs can own space on a volume, but only one catalog may *reside* on a volume.

You can assign data space to a catalog in one of the following ways, by specifying:

- The DEDICATE parameter (at the catalog level only), an optional space allocation parameter(s) (at the data or index component levels), and a DLBL job control statement (required for the master catalog only).
- The ORIGIN parameter, a space allocation parameter(s), and a DLBL job control statement (required for the master catalog only). Note that the first track (minimum CA) on a volume is never specified in an ORIGIN parameter. One reason is because a portion of the first cylinder contains the IPL record and therefore is not available for your use.
- A space allocation parameter(s), no DEDICATE or ORIGIN parameters, and a DLBL job control statement (required for the master catalog only).

Note: When you define a catalog, you can allocate space for both its data and index component. However, VSE/VSAM calculates the space required for the catalog's index component depending on your data allocation. Therefore, your index allocation may be overridden by VSE/VSAM, especially when you allocate more than the maximum space required.

Using the DEDICATE Parameter for a Catalog

You use the DEDICATE parameter to specify that all the unowned and unallocated (free) space on a specified volume is to belong to a catalog. Note that VSE/VSAM acquires all the free space for the catalog if 16 or fewer extents exist on the volume (where extents are contiguous areas of free space). If more than 16 "free" extents exist on the volume, VSE/VSAM acquires only the first 16 extents for the catalog.

You can specify DEDICATE at the catalog level only. The specification is mutually exclusive with the space allocation parameters (CYLINDERS, BLOCKS, RECORDS, TRACKS). The amount of space to be suballocated to the catalog itself, depends on how (or if) you specify a space allocation parameter at the index component and/or data component levels.

If you are defining a master catalog, a DLBL job control statement is required. The statement can appear in the job stream, or it can reside on the permanent standard label area.

Examples for "DEDICATE":

- Specify DEDICATE and no space allocation parameter at the data and index component levels (EXAMPLE 1). In this case, the acquired free space on the designated volume is owned by the catalog. The amount of space suballocated to the catalog itself is calculated by VSE/VSAM so as to provide enough space for a minimum size catalog. This amount is device-dependent. The remaining space is available for other VSE/VSAM objects.

```
// JOB  EXAMPLE 1
// DLBL IJSYSCT,'VSAM.MASTER.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
   DEFINE MASTERCATALOG              -
          (NAME(VSAM.MASTER.CATALOG) -
           VOLUME(DOSRES)            -
           DEDICATE)
/*
/&
```

- Specify DEDICATE and a space allocation parameter at the data and index component levels (EXAMPLE 2). In this case, the acquired free space on the designated volume is owned by the catalog, and the space suballocated to the catalog is the sum of the data and index level specifications. The space not suballocated to the catalog (if any) is available for other VSE/VSAM objects.

```
// JOB  EXAMPLE 2
// EXEC IDCAMS,SIZE=AUTO
   DEFINE USERCATALOG             -
          (NAME(VSE.USER.CATALOG) -
           VOLUME(USRVOL)         -
           DEDICATE)              -
          DATA(CYLINDERS(10))     -
          INDEX(CYLINDERS(2))
/*
/&
```

- Specify DEDICATE and a space allocation parameter at the data component level (EXAMPLE 3). In this case, the acquired free space on the designated volume is owned by the catalog. VSE/VSAM uses your

data component specification to calculate a value for the index component and it then adds this value to the data component specification. This sum becomes the amount of data space that is suballocated to the catalog. The space not suballocated to the catalog (if any) is available for other VSE/VSAM objects.

```
// JOB  EXAMPLE 3
// DLBL IJSYSCT,'VSAM.MASTER.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
   DEFINE MASTERCATALOG              -
          (NAME(VSAM.MASTER.CATALOG) -
           VOLUME(DOSRES)            -
           DEDICATE)                 -
          DATA(CYLINDERS(6))
/*
/&
```

Using the ORIGIN Parameter for a Catalog

You use the ORIGIN parameter to specify the beginning point (track number or block number) of the catalog's data space. VSE/VSAM determines the ending point of the data space from the value you specify (at the catalog level) for CYLINDERS, BLOCKS, RECORDS, or TRACKS (VSE/VSAM converts records to tracks). For BIG-DASD and FAT-DASD, value of the ORIGIN parameter should be in multiples of 15, because these types of DASD are positioned at cylinder boundaries while the ORIGIN parameter is specified in tracks. In the examples below, using Small DASD is assumed; this does not impose restrictions onto the ORIGIN parameter.

- If you specify a block number value (in the ORIGIN parameter) that does not coincide with a minimum CA (track) boundary, VSE/VSAM rounds the value up to the next minimum CA boundary.
- If the ending block value (ORIGIN plus number of blocks) does not coincide with a minimum CA boundary, VSE/VSAM rounds it down to the previous minimum CA boundary.
- For a SCSI device, explicit space allocation (at least 3072 blocks) is required.

You can specify ORIGIN at the catalog level only. If you are defining a master catalog, a DLBL statement is required. It can reside on the permanent standard label area or it can appear in the job stream.

Examples for "ORIGIN":

- Specify ORIGIN and a space allocation parameter at the catalog level only (EXAMPLE 4). In this case, VSE/VSAM divides the amount of space you specified between the catalog's data and index components. The entire amount of space is owned by the catalog and is available only to the catalog.

```
// JOB  EXAMPLE 4
// EXEC IDCAMS,SIZE=AUTO
   DEFINE USERCATALOG             -
          (NAME(VSE.USER.CATALOG) -
           VOLUME(USRVOL)         -
           ORIGIN(20)             -
           TRACKS(260))
/*
/&
```

- Specify ORIGIN at the catalog level only and a space allocation parameter at the catalog, data component, and index component levels (EXAMPLE 5). In this case, VSE/VSAM adds the data component value to the index component value to arrive at the total amount of space to be suballocated to the catalog. The sum cannot be greater than the space allocation value specified at the catalog level. If the sum is less than the catalog level specification, the remainder is available for other VSE/VSAM objects.

```
// JOB  EXAMPLE 5
// DLBL IJSYSCT,'VSAM.MASTER.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
   DEFINE MASTERCATALOG              -
          (NAME(VSAM.MASTER.CATALOG) -
           VOLUME(DOSRES)            -
           ORIGIN(20)                -
           TRACKS(60))               -
          DATA(TRACKS(40))           -
          INDEX(TRACKS(20))
```

```
/*
/&
```

- Specify ORIGIN at the catalog level only and a space allocation parameter at the catalog and data component levels (EXAMPLE 6). In this case, VSE/VSAM uses your data component specification to calculate a value for the index component. Note that it is your responsibility to provide enough space for the index component. You can do this through the catalog level specification. VSE/VSAM then adds the calculated value to the value specified in the data component. This amount is then suballocated to the catalog. The space cannot be greater than the space allocation value specified at the catalog level. If the sum is less than the catalog level specification, the remainder is available for other VSE/VSAM objects.

```
// JOB  EXAMPLE 6
// EXEC IDCAMS,SIZE=AUTO
   DEFINE USERCATALOG             -
          (NAME(VSE.USER.CATALOG) -
           VOLUME(USRVOL)         -
           ORIGIN(20)             -
           TRACKS(80))            -
          DATA(TRACKS(40))
/*
/&
```

Using neither DEDICATE nor ORIGIN for a Catalog

By *not* specifying the DEDICATE or ORIGIN parameters (EXAMPLE 7), you indicate that you want VSE/VSAM to choose the first available extent on the volume that is large enough to contain your primary allocation (VSE/VSAM in effect, defaults to ORIGIN). The space allocation parameter (TRACKS, CYLINDERS, RECORDS, or BLOCKS) determines the data space allocation for the VSE/VSAM catalog.

If you are defining a master catalog, a DLBL job control statement is required. It can reside on the permanent standard label area, or it can appear in the job stream.

```
// JOB  EXAMPLE 7
// EXEC IDCAMS,SIZE=AUTO
   DEFINE USERCATALOG             -
          (NAME(VSE.USER.CATALOG) -
           VOLUME(USRVOL)         -
           TRACKS(260))
/*
/&
```

Defining a VSE/VSAM Data Space

The DEFINE SPACE command is used to define VSE/VSAM data spaces, or to reserve a volume for VSE/VSAM's future use.

VSE/VSAM Data Spaces on a Volume

A VSE/VSAM data space is space on a disk volume that is owned and managed by VSE/VSAM. When you define a data space on a volume (or you specify the volume as a candidate to contain VSE/VSAM objects), you are, in effect, giving control to the catalog in which the data space was defined. The catalog now "owns" the data space and will suballocate portions of that space to VSE/VSAM objects.

A data space can take up all or part of the space on one volume; it cannot take up space on several volumes. A volume can contain several data spaces.

When you define a data space with the DEFINE SPACE command, you must specify the volume that is to contain the data space. IDCAMS creates a volume entry in the catalog to describe every volume on which one or more data spaces have been defined. The volume on which data space is to be defined must be mounted for the DEFINE.

If there is space available, VSE/VSAM ownership is indicated in the volume's VTOC with a format-4 label and a VSE/VSAM data space is allocated on the volume.

You can assign space to one of eight performance classes, that is, you can classify space as standard (nonfixed head) space, fixed-head space, or whatever other space criteria you wish to choose.

VSE/VSAM Objects in a Data Space

A data space can contain several suballocated files, and a file can be stored in several data spaces, on the same or on different volumes.

Space Allocation to VSE/VSAM Objects

VSE/VSAM suballocates space for a file from space that is available in an existing data space. Data spaces cannot be dynamically extended in VSE/VSAM (this applies to unique files and their associated data space). You must define more space (or free existing space) whenever a DEFINE or EXTEND fails because of insufficient space. If all of the space given to a unique file at DEFINE is used, the unique file cannot be further extended, even by the definition of new data spaces.

The following examples show how you can define a data space. (See also “Example 2: Define a VSE/VSAM User Catalog and a VSE/VSAM Data Space” on page 210.) For the examples assume that:

- The data space is to be cataloged in the master catalog.
- The master catalog's DLBL statement is in the standard label area.
- Standard format-1 and format-3 labels describing the data space are written into the volume's VTOC.

EXAMPLE 8 specifies that 209 tracks, beginning with track 19, are to be allocated to VSE/VSAM. Small DASD volume is assumed here; in case of Large DASD, parameter ORIGIN should have a different value. (See “Using the ORIGIN Parameter for a Catalog” on page 18.)

```
// JOB  EXAMPLE 8
// EXEC IDCAMS,SIZE=AUTO
   DEFINE SPACE           -
          (TRACKS(209)    -
           VOLUME(VOLS12) -
           ORIGIN(19))    -
          CATALOG(VSAM.MASTER.CATALOG/UPDPW1)
/*
/&
```

EXAMPLE 9 is a "default ORIGIN" example. By not providing the ORIGIN (or DEDICATE) parameter you cause VSE/VSAM to choose the first possible extent on the volume that can satisfy your space allocation quantity (in this case, 209 tracks).

```
// JOB  EXAMPLE 9
// EXEC IDCAMS,SIZE=AUTO
   DEFINE SPACE            -
          (TRACKS(209)     -
           VOLUME(VOLS12)) -
          CATALOG(VSAM.MASTER.CATALOG)
/*
/&
```

EXAMPLE 10 indicates that the unowned and unallocated data space (maximum of 16 extents per volume) on volume VSER01 is to be allocated to VSE/VSAM.

```
// JOB  EXAMPLE 10
// EXEC IDCAMS,SIZE=AUTO
   DEFINE SPACE            -
          (DEDICATE        -
           VOLUME(VSER01)) -
          CATALOG(VSAM.MASTER.CATALOG/UPDPW1)
/*
/&
```

EXAMPLE 11 The parameter FATDASD allocates every available cylinder up to 65520 on volume VSER01.

```
// JOB  EXAMPLE 11
// EXEC IDCAMS,SIZE=AUTO
   DEFINE SPACE                    -
          (DEDICATE                -
```

```
            FATDASD                     -
            VOLUME(VSER01))             -
           CATALOG(VSAM.MASTER.CATALOG/UPDPW1)
/*
/&
```

Distributed Free Space

When you define a key-sequenced file or an alternate index, you can request (through the FREESPACE parameter) that a certain percentage of empty space (called *free space*) is to be distributed throughout the file:

- Upon the initial load of the file and every subsequent extension, and
- When many records are inserted at the same time.

VSE/VSAM can subsequently use this free space to insert new records or lengthen existing records. When the space you originally designated as *free* is used up by VSE/VSAM and more is needed, a *splitting* of the CI (control interval) or CA (control area) occurs. That is, VSE/VSAM automatically acquires a new CI or CA (whichever is applicable) and divides the records in the original CI or CA between the old and the new.

Because splitting takes time and degrades performance, it is important that you try to match the anticipated file change activity with a meaningful free-space specification. (Refer to "Hints on Specifying Free Space" on page 21.) Note that an excessive amount of free space wastes disk storage.

You can cause free space to be initially distributed within a file by specifying one of the following:

- Some space is to be left free at the end of every used CI
- Some CIs in every CA are to be left completely empty
- A combination of the above two

You specify free space for both the CI and the CA as a percentage of the total space for the respective unit. For example:

```
   FREESPACE(20 10)
```

This indicates that 20% of the space in every CI is to be initially empty and 10% of every CA is to be initially empty.

If you specify a CI percentage that is more than zero but less than the maximum logical record size, VSE/VSAM may not reserve enough space in the CI to contain a logical record. If you specify a CA percentage that is more than zero but less than one CI, VSE/VSAM reserves one CI for every CA. The system default for free space is 0.

If your LISTCAT output shows that excessive splits are occurring, you should consider reorganizing your file. You can use the REPRO command to reorganize your file.

Hints on Specifying Free Space

If you know that additions are to occur only in a specific part of the file:

1. Load the parts that are not to receive additions with a free-space specification of (0 0).
2. After you have loaded that portion of the file:
   a. Close the file
   b. Change the free space specification (through the ALTER command) to the value you want
   c. Load those parts of the file that are to receive the additions

By this method, you can easily attain different free-space specifications for various parts of the file.

If you know that additions are to occur throughout a file, but the additions are unevenly distributed, specify a small amount of CA free space when you define the file. Specify just enough to hold the average number of insertions expected within the CA before the file is reorganized. If VSE/VSAM finds insufficient

space in some CAs, new CAs will be created through the split process. The new CAs will contain free CIs which are used for additional insertions in the vicinity. Using this method, you gain two advantages:

- Additional splits (after the first split) in the part of the file with the most growth are minimized.
- You save storage space because the CIs that have little or no growth do not contain unneeded free space.

If there are to be few additions to the file, consider a free-space specification of (0 0) for loading the file. When records are added, new CAs are created at the point of insertion to provide room for additional insertions in the vicinity. In this case, unused free space is not provided.

For random (direct) processing insertions which are fairly uniformly distributed across the file, specify CI free space sufficient to contain most of the expected insertions in an average CI during the time before the file is reorganized. Control area (CA) free space should be sufficient to handle those cases where there are extra insertions in some control intervals (CIs).

If the insertion frequency can be estimated, this approach can allow you to allocate sufficient free space for significant numbers of insertions without excessive CA splits.

Defining a VSE/VSAM File (Cluster)

Defining VSE/VSAM Files

A VSE/VSAM file can be:

- Suballocated. A suballocated file shares a data space with other files
- Not allocated (dynamic). An unallocated (dynamic) file has no space allocated to it at define time.
- Unique. A unique file has a data space to itself.

To define a *suballocated* VSE/VSAM file, you first define a data space, then use the DEFINE CLUSTER command. VSE/VSAM suballocates space for the file in the data space you have set up and enters information about the file in a VSE/VSAM catalog. (Note that at this time, no records are loaded into the file. Defining a file is distinct from loading records into it.) A file can be stored in several data spaces on the same or different volumes. For more information and examples, see “Defining a Suballocated VSE/VSAM File” on page 23.

To define a *dynamic* VSE/VSAM file, you specify the NOALLOCATION and REUSE parameters at define time. The required space (specified with a space allocation parameter at define time) is suballocated to the file when VSE/VSAM opens it. For more information about dynamic files, refer to "NOALLOCATION" in the VSE/VSAM User's Guide and Application Programming.

To define a *unique* VSE/VSAM file, you do *not* define the data space beforehand. Instead, you have to:

1. Specify the parameter UNIQUE in the DEFINE CLUSTER command.
2. Assign space to the file with a space allocation parameter and the DLBL/EXTENT job control statements.

The data space is acquired and assigned to the file concurrent with the file definition. The volume(s) to contain a unique file must be mounted as in defining a data space. For more information and examples, see “Defining a Unique VSE/VSAM File” on page 24.

An exception is the definition of a unique VRDS file. Only the data object of a VRDS file can be defined as unique file. The index object (internally created by VSE/VSAM) needs suballocation data space. The VOLUMES parameter of the 'cluster component' has to point to a volume containing suballocation data space.

With a key-sequenced file on several volumes, you may assign data to various volumes according to ranges of key values. For example, if you have three volumes, you might assign records with keys A-E to the first volume, F-M to the second, and N-Z to the third. (The keys could also be A-D, G-K, L-O, R-W, and so on.)

About Files and Clusters

VSE/VSAM treats all files as clusters. A cluster consists of:

- A data component only

  (in the case of an entry-sequenced file or a relative-record file),
  or

- A data component and an index component

  (in the case of a key-sequenced file).

Besides setting up a catalog entry for every component of a cluster, VSE/VSAM sets up a catalog entry for the cluster as a whole. This entry consists mainly of the name (the 44-byte file ID) of the cluster which you specify in the DEFINE command, and, for a key-sequenced file, an indication of the relationship between the data component and the index component.

You can also specify names for the index and data components of a cluster in the INDEX and DATA parameters of the DEFINE CLUSTER command. (If you do not provide names for the data and index components, VSE/VSAM generates them.) These names enable you to process every component individually. For example, you may open the index of a key-sequenced file separately and process it as data (with addressed or CI access).

You can also specify that the data and index components of a key-sequenced file are to reside on different volumes. You do this by specifying the VOLUMES parameter as an attribute of both DATA and INDEX.

A cluster is defined in the master catalog unless you specify a job catalog or indicate otherwise through the CATALOG parameter.

Specifying Information that Defines a File

When a DEFINE command is used to define a key-sequenced cluster, three entries are created in the catalog: an entry for the cluster, its data component, and its index component.

Attributes of the data and index components can be specified separately from the attributes of the cluster as a whole. If attributes are specified for the cluster as a whole and are not specified for the components, the attributes of the cluster (except for its passwords and other protection attributes) apply to its components. If an attribute that is applicable to the data or index component is specified for both the cluster and the component, the component specification overrides the cluster specification.

For information on which parameters can be used with every type of entry, see “DEFINE CLUSTER” on page 97.

The following examples demonstrate how to use the basic DEFINE parameters to define a file without considering all of the options. See also “Example 3: Define VSE/VSAM Files” on page 212 and “Example 4: Define NonVSAM and VSE/VSAM Files” on page 215.

Defining a Suballocated VSE/VSAM File

When a VSE/VSAM file is defined and space is suballocated for it in one or more existing data spaces, DLBL and EXTENT statements are not required and the volume(s) on which the file is defined need not be mounted.

EXAMPLE 11 shows how to define a suballocated key-sequenced file. The cluster is defined in the master catalog. (No DLBL job control statement is shown for the master catalog, because it is assumed that it was previously entered in the standard label area.)

```
// JOB  EXAMPLE 11
// EXEC IDCAMS,SIZE=AUTO

   DEFINE  CLUSTER(             -
             NAME(MSTRFILE)     -
             VOLUMES(VOLS12)    -
             TRACKS(38 19)      -
```

```
              KEYS(10 1)          -
              RECORDSIZE(80 80)) -
            CATALOG(VSAM.MASTER.CATALOG/UPDPW1)
/*
/&
```

The volume on which the file will reside is indicated in the VOLUME parameter. The TRACKS, CYLINDERS, BLOCKS, or RECORDS parameter indicates how the space is allocated:

Initially (primary allocation)
Optionally, if the file must be extended (secondary allocation)

VSE/VSAM selects the data space(s) on a volume from which to suballocate space to the file. All volumes must be of the same type and capacity for every component of a file. If several volumes are specified, the additional volumes can be used when the file is extended. These volumes are described in the file's catalog entry as potential *candidate* volumes.

The length and offset of a cluster's key-field (KEYS) and the average and maximum length of data records (RECORDSIZE) is also specified in EXAMPLE 11.

Defining a Unique VSE/VSAM File

A file can be defined at the same time as the data space(s) which will contain it. In this case, the file is called *unique* and no other file can occupy its data space(s).

The data and the index of a key-sequenced unique file will occupy separate data spaces. Every component requires DLBL and EXTENT statements if the UNIQUE option is specified as part of the cluster definition or both components are defined as unique individually. An entry-sequenced file or relative record file occupy only one data space which also requires a DLBL and an EXTENT statement.

A variable-length relative-record data set (VRDS) occupies only a unique data space for the data component. The index component administrated internally by VSE/VSAM will reside in suballocated VSE/VSAM space on the same disk.

EXAMPLE 12 assumes that a unique key-sequenced file is defined in the master catalog. The specifications in the example cause the creation of four entries: a volume, cluster, data, and index entry.

```
// JOB  EXAMPLE 12
// DLBL DFILE,,,VSAM
// EXTENT  ,VOLS12,,,19,19
// DLBL    XFILE,,,VSAM
// EXTENT  ,VOLS12,,,38,19
// EXEC    IDCAMS,SIZE=AUTO
   DEFINE  CLUSTER(NAME(PAYROLL1)   -
            RDPW(DEPT27R)           -
            VOL(VOLS12)             -
            RECORDSIZE(100 475)     -
            KEYS(12 4))             -
          DATA(NAME(PAYROLL1.DATA)   -
            UNIQUE                  -
            CYLINDERS(1 1)          -
            FILE(DFILE))            -
          INDEX(NAME(PAYROLL1.INDEX) -
            UNIQUE                  -
            FILE(XFILE)             -
            CYLINDERS(1 1))         -
          CATALOG(VSAM.MASTER.CATALOG/UPDPW1)
/*
/&
```

For a key-sequenced file with the UNIQUE attribute where the data and index components reside on the same volume, the FILE parameter must be specified under both DATA and INDEX. The VOLUMES parameter and the space allocation parameter (CYLINDERS, BLOCKS, TRACKS, or RECORDS) must be included in the DEFINE command (unless you specified the MODEL parameter).

If you specify the size of the area allocated to a unique index (as is done in this example), you must ensure that it is large enough.

The space allocated to the data in a unique file:

- For CKD devices: must be an integral number of cylinders and every extent must begin and end on cylinder boundaries.
- For FBA devices: must begin and end on a minimum CA boundary and must be a whole number of minimum CAs. For space allocation, use the BLOCKS parameter. (For a discussion of minimum CAs and maximum CAs, refer to the "Control Area Size" in the VSE/VSAM User's Guide and Application Programming.)
  - On ECKD devices, unique datasets (CLUSTER or AIX) must not allocate mixed extents of FAT-DASD and non FAT-DASD. The mix of Small and BIG-DASDs (3390 mod 9/27) (max. 10017 cylinders) is still allowed and not affected.
  - In order to define a CLUSTER or AIX as UNIQUE with option FAT-DASD, make sure that each volume used for the assigned extents
    - either is predefined as FAT-DASD to the current VSAM catalog,
    - or has a minimum real capacity of 64K+1 tracks.

A unique file can have a maximum of 16 extents per volume, but it cannot be extended, and space left over after the records are loaded cannot be released.

Defining a Key-Sequenced File

EXAMPLE 13 shows the DEFINE command for setting up a key-sequenced cluster that consists of a data component and an index component. The CLUSTER, DATA, and INDEX parameters are all specified, so the data and the index components of the cluster can be explicitly named rather than letting VSE/VSAM name them. (Note that at this time, no records are loaded into the file. Defining a file is distinct from loading records into it. For an example of how to load a file, see “Loading Records into a File” on page 31.)

In the example, the specifications:

- Provides space for 10,000 data records (fixed-length, 250 bytes) on every key range. Any future extensions are to be made in increments of space for 500 records. Keys are 15 bytes long and begin in the thirty-first byte (defined as relative position 30, because byte 1 is relative position 0) of the records.
- Free space is to be 20% of every CI and 10% of every CA. Refer to "Distributed Free Space" in the VSE/VSAM User's Guide and Application Programming.
- Sequence-set index records are to be placed adjacent to CAs in the file.

Assume that the cluster is to be defined in the master catalog and be in suballocated space on volumes VOLS0A and VOLS0B. These volumes need not be mounted, because VSE/VSAM can determine whether and what space is available merely by examining the master catalog, which owns these volumes.

```
// JOB   EXAMPLE 13
// EXEC  IDCAMS,SIZE=AUTO
   DEFINE  CLUSTER                  -
             (NAME(MYKSDS)          -
             VOLUMES(VOLS0A VOLS0B) -
             KEYRANGES((A M) (N Z)) -
             RECORDS(10000 500)     -
             ORDERED)               -
           DATA                     -
             (NAME(MYDATA)          -
             KEYS(15 30)            -
             RECORDSIZE(250 250)    -
             BUFFERSPACE(8192)      -
             FREESPACE(20 10))      -
           INDEX                    -
             (NAME(MYINDEX))        -
             CATALOG(VSAM.MASTER.CATALOG/UPDPW1)
/*
/&
```

The following provides further explanations to EXAMPLE 13 and parameters.

The INDEXED Parameter

Indicates that a key-sequenced cluster is to be defined. The parameter is the default and does, therefore, not need to be specified.

The RECORDS Parameter

Indicates primary and secondary allocation quantities. Specifying the number of records, independent of the number of physical units, such as blocks, tracks or cylinders, leaves the calculation of the number of physical units of space up to VSE/VSAM. VSE/VSAM calculates the size of the CI and CA to be used. You may specify the CI size yourself, and VSE/VSAM will use it as long as it falls within the acceptable limits that VSE/VSAM calculates.

The BUFFERSPACE Parameter

specifies the smallest amount of virtual storage a processing program will ever provide for I/O buffers to process MYKSDS. A multiple of 512 should, however, be specified to avoid wasting space. If you do not specify BUFFERSPACE, VSE/VSAM determines CI size first and then sets buffer space equal to the size of two data CIs plus one index CI.

If the values you specify for record length and key length require CIs too large for the buffer space you specify, your DEFINE will fail.

The relationship between CI size and least amount of I/O buffer space is further discussed in "Optimizing the Performance of VSE/VSAM" in the VSE/VSAM User's Guide and Application Programming.

The KEYRANGES Parameter

With the parameter, you may assign data to the various volumes of a key-sequenced file according to ranges of key values. For example, if you have three volumes you might assign records with keys A-E to the first volume, F-M to the second, and N-Z to the third.

The amount of space specified in the primary allocation parameter is allocated to *every* key range. If the number of volumes is:

- Higher than the number of key ranges, the excess volumes will become candidate volumes for all the key ranges.
- Lower than the key ranges, the key ranges collect on the last volume.

If the values specified for the KEYRANGES sub-parameters are shorter than those specified for the actual keys, then IDCAMS pads the low key range to the right with binary zeros and the high key range to the right with binary ones. However, the low key (and the high key) must be of the same length in all key ranges.

The VOLUMES, ORDERED, CYLINDERS, and KEYRANGES Parameters

The following examples (13A through 13F) illustrate the use of the parameters.

EXAMPLE 13A:

```
    VOLUMES (A B C)
    KEYRANGES ( (00 30) (31 65) (66 99) )
    ORDERED
    CYLINDERS (100 10)
```

A primary allocation of 100 cylinders will be made for *every* key range. The first key range will be on volume A, the second on B, and the third on C. If 100 cylinders cannot be allocated on every volume, the request is rejected. A key range can be extended only on the volume it occupies or on a candidate volume. Thus, if volume D were added to the list, all key ranges will be extended on volume D if the appropriate volume initially assigned to the key range is full. If only volumes A and B were specified, the first key

range would be allocated on volume A and the second and third key ranges would be allocated on volume B. EXAMPLE 13B:

```
VOLUMES (A B C)
KEYRANGES ( (00 30) (31 65) (66 99) )
UNORDERED
CYLINDERS (50 5)
```

A primary allocation of 50 cylinders will be made for *every* key range. VSE/VSAM attempts to put one key range on every volume. If volume A does not have 50 cylinders available, the first key range is put on volume B and the second and third on volume C. If neither A nor B has 50 available cylinders, all three key ranges are placed on volume C. A key range will be extended first on the volume it is on, then it will be extended on any candidate volume. A candidate volume is a volume that is named in the volume list for a keyrange file but was not initially assigned to a key range, that is, there may be more volumes than key ranges in the list. However, if volume D were available as a candidate volume, every key range would be extended on volume D if no more space were available on the volume of its primary allocation. A key range can cover the volume of primary allocation and any candidate volume.

The ORDERED parameter indicates that space must be suballocated on the volumes in the order in which they are listed in the VOLUMES parameter. In particular, for key-ranged files, ORDERED forces primary allocation for the first key range to be made on the first volume of the volume list, primary allocation for the second key range to be made on the second volume of the volume list, and so on. If a volume cannot accommodate the space for the appropriate primary allocation, the DEFINE fails.

Thus, in the job EXAMPLE 13 above, VOLS0A has 10,000 records allocated for key range A-M, and VOLS0B has 10,000 records allocated for the key range N-Z. If VOLS0A does not have enough space to accommodate the 10,000 records in the primary allocation, the DEFINE fails. Contrast this to the UNORDERED case in which both key ranges would be on VOLS0B provided VOLS0B has enough space, and VOLS0A would then become a candidate volume.

Space on every volume could also be specified in the number of records, even with variable-length records: VSE/VSAM uses the average size (250 bytes) to calculate the number of cylinders for every volume.

The following examples (13C through 13F) further illustrate the use of the VOLUMES and ORDERED parameters.

EXAMPLE 13C:

```
VOLUMES (A B C)
ORDERED
CYLINDERS (50 5)
```

The 50 cylinders of primary space for the file must be available on volume A, or the request will be rejected. Volumes B and C are candidate volumes. If the file is extended, a five cylinder secondary space allocation is made on volume A if it has enough data space. Otherwise, a primary allocation of 50 cylinders is made on volume B. If volume B does not have enough data space for a primary allocation, the request for extension is rejected. When the file is subsequently extended, the secondary allocations are made on volume B if it has enough data space. Otherwise, a primary allocation is made on volume C.

EXAMPLE 13D:

```
VOLUMES (A B C)
UNORDERED
CYLINDERS (50 5)
```

The 50 cylinder primary allocation for the file can be made on either volume A, B, or C. However, if all 50 cylinders cannot be allocated on one volume, the request is rejected. The volumes are searched in the order they are specified. If both volumes A and B have 50 cylinders available, the allocation will be made on volume A. If the file is extended, the five cylinder secondary allocations are made on the volume the last primary allocation was made on until the volume is full. However, the first allocation to be made on any volume is always a primary allocation of 50 cylinders. Once again, the volumes are searched for space in the order specified.

EXAMPLE 13E:

```
   VOLUMES (A B C)
   ORDERED
   CYLINDERS (50)
```

The 50 cylinders of primary space for the file must be available on volume A, or the request will be rejected. Volumes B and C are candidate volumes. If the file is extended, a primary allocation of 50 cylinders is made on volume B. If volume B does not have enough data space for a primary allocation, the request for extension is rejected. When the file is subsequently extended, a primary allocation is made on volume C.

EXAMPLE 13F:

```
   VOLUMES (A B C)
   ORDERED
   CYLINDERS (1000 10)
```

This request will be rejected because the amount of primary space to be allocated on every volume is greater than one volume. The primary allocation *must* be small enough to be satisfied by one volume.

Defining an Extra-Large Dataset

Extra-large datasets are supported by the following VSE/VSAM dialogs:

- Display and Process a File
- Define a New File

The dialogs display additional selection information as follows:

```
FILE ADDRESSABILITY:   2    1=NOT EXTENDED    2=EXTENDED  (KSDS only)
```

Figure 1 on page 28 shows the panel for defining an extra-large dataset.

```
 IESFILDEFA                    DEFINE A NEW FILE

 CATALOG NAME:          VSESPUC

 FILE ID.............. VSAM____ . KSDS____ . FOUR____ . GIGA____ . ________
 FILE NAME............ KSDS4GB

 FILE ORGANIZATION..... 2          1=Non keyed  (ESDS)  3=Numbered   (RRDS)
                                   2=Keyed      (KSDS)  4=Numbered   (VRDS)
                                                        5=Sequential (SAMESDS)
 FILE ADDRESSABILITY... 2          1=NOT EXTENDED       2=EXTENDED  (KSDS only)

 FILE ACCESS.......... 1          1=Multiple Read OR Single Write
                                  2=Multiple Read AND Single Write
                                  3=Multiple Read AND Write (no integrity)
                                  4=Multiple Read AND Write (with integrity)

 FILE USAGE........... 1          1=File is used as a Data File (NOREUSE)
                                  2=File is used as a Work File (REUSE)

PF1=HELP        2=REDISPLAY  3=END        4=RETURN
```

Figure 1. Defining an Extra-Large Dataset

The dialog accepts extended addressing only if the following has been specified: 2 (Keyed KSDS) for FILE ORGANIZATION and 1 (NOREUSE) for FILE USAGE.

Following is a job stream example created by the dialog for a new file defined with extended addressing.

```
* $$ JOB JNM=KSDS4GB,CLASS=0,DISP=D,NTFY=YES
// JOB  KS4  DEFINE FILE
// EXEC IDCAMS,SIZE=AUTO
 DEFINE CLUSTER (                                -
        NAME (VSAM.KSDS.FOUR.GIGA)               -
        CYLINDERS(4000     3000)                 -
        SHAREOPTIONS (1)                         -
        RECORDSIZE (80     80)                   -
        VOLUMES (DOSRES SYSWK1)                  -
        NOREUSE                                  -
        INDEXED                                  -
        FREESPACE (15 7)                         -
        KEYS (12  0)                             -
        NOCOMPRESSED                             -
        EXTRALARGEDATASET                        -
        TO (99366  ))                            -
        DATA (NAME (VSAM.KSDS.FOUR.GIGA.@D@)     -
        CONTROLINTERVALSIZE (4096))              -
        INDEX (NAME (VSAM.KSDS.FOUR.GIGA.@I@)) -
        CATALOG (VSESP.USER.CATALOG)
  IF LASTCC NE 0 THEN CANCEL JOB
/*
// OPTION STDLABEL=ADD
// DLBL KSDS4GB,'VSAM.KSDS.FOUR.GIGA',,VSAM,                          X
               CAT=VSESPUC
/*
// EXEC IESVCLUP,SIZE=AUTO
A VSAM.KSDS.FOUR.GIGA                          KSDS4GB VSESPUC
/*
/&
* $$ EOJ
```

Figure 2. Job Stream Example for Defining a KSDS File with Extended Addressing

Note: To convert an existing KSDS for extended addressing and thus permit it to grow above 4 GB, proceed as follows:

1. REPRO the existing KSDS (to tape or disk).
2. Delete the existing KSDS.
3. Define the new KSDS as described in this section.
4. REPRO (restore the records saved in step 1).

Defining an Entry-Sequenced File

Though the data component of an entry-sequenced file is the only member of its cluster, you must define a cluster for it.

EXAMPLE 14 shows that records in the file (a cluster named ENTRY) are to be stored on three volumes: USRVOL, UVOLSB, UVOLSC. Note that before starting this job, you have to allocate space on all of these three volumes for the catalog VSE.USER.CATALOG using the DEFINE SPACE command.

You indicate that the cluster is to be defined in job catalog VSE.USER.CATALOG by specifying a IJSYSUC DLBL statement with file identifier VSE.USER.CATALOG. Records are of variable length, up to 700 bytes, with an average size of 500 bytes. (For an alternate way of specifying a cluster's catalog, refer to EXAMPLE 15.)

```
   // JOB   EXAMPLE 14
   // DLBL  IJSYSUC,'VSE.USER.CATALOG',,VSAM
   // EXEC  IDCAMS,SIZE=AUTO
      DEFINE  CLUSTER-
                (NAME(ENTRY)                  -
                VOLUMES(USRVOL UVOLSB UVOLSC) -
                ORDERED                       -
                CYLINDERS(10 2)               -
                RECORDSIZE(500 700)           -
                BUFFERSPACE(5120)             -
                NONINDEXED)                   -
              DATA                            -
                (NAME(ENTRY.DATA))
   /*
   /&
```

The NONINDEXED parameter indicates that an entry-sequenced cluster is to be defined.

The primary allocation of 10 cylinders is only made on volume USRVOL. The additional volumes are used only when the file is extended. These volumes are described in the file's catalog entry as candidate volumes.

Defining a Relative-Record File

Like an entry-sequenced file, the data component of a relative-record file is the only member of its cluster. A relative-record file can be seen as a string of fixed-length slots or record areas, every one of which is assigned a relative-record number, starting from 1.

EXAMPLE 15 shows that records in the file (a cluster named STOCKINV) be stored on a 3375 volume.

The cluster is to be defined in a catalog other than the default catalog. Therefore, you must use the CATALOG parameter to indicate the correct catalog (VSE.USER.CATALOG). The records in the file have a fixed length of 132 bytes (the average and maximum record sizes must be equal for a relative-record file). The NUMBERED parameter indicates that a relative-record file is to be defined. The primary number of tracks allocated is 50, and ten secondary tracks are allowed for every extension. Extensions must be suballocated on the volume specified as UVOLSB (that is, the same volume that contains the primary tracks).

```
// JOB   EXAMPLE 15
// EXEC  IDCAMS,SIZE=AUTO
   DEFINE  CLUSTER(              -
             NAME(STOCKINV)      -
             VOL(UVOLSB)         -
             TRACKS(50 10)       -
             RECORDSIZE(132 132) -
             NUMBERED)           -
           CATALOG(VSE.USER.CATALOG)
/*
/&
```

Defining a Variable Length Relative-Record File

For the application, a VRDS file appears like an RRDS file. However, internally VSE/VSAM maintains a small index and the VRDS looks like a KSDS. (LISTCAT shows the index object.)

The following example shows the definition of a VRDS. In the example:

- NUMBERED is defined together with RECORDSIZE(80 2000). This results in the definition of a VRDS.
- Because (as with a KSDS) inserts between existing records could force CI or CA splits, a FREESPACE for the internal index object is accepted.

```
// JOB DEFINE CLUSTER VRDS
// DLBL IJSYSUC,'VSE.USER.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
   DEFINE  CLUSTER-
             (NAME(TEST.VRDS)      -
             VOLUMES(444444 )      -
             NUMBERED              -
             RECORDSIZE(80  2000)  -
             FREESPACE(20 20)      -
             CYL  (1  1)           -
             CISZ(2048)            -
                  )                -
             DATA (NAME(TEST.VRDS.DATA))
/*
/&
```

Defining a UNIQUE Variable Relative-Record File

The following example shows the definition of a unique VRDS. Only the data component can be unique. The internally created index component requires suballocation data space. The volume on which the

index will reside is indicated in the VOLUMES parameter at the "cluster level" of the DEFINE CLUSTER command.

```
// JOB DEFINE UNIQUE VRDS CLUSTER
// DLBL IJSYSUC,'VSE.USER.CATALOG',,VSAM
// DLBL VRDSDAT,,,VSAM              DATA PART OF VRDS
// EXTENT ,666666,,,60,15           EXTENTS OF VRDS DATA
// EXEC IDCAMS,SIZE=AUTO
 DEFINE CLUSTER                   -
        (NAME(TESTU)              -
        NUMBERED                  -
        RECSZ ( 80  2000)         -
        VOLUMES (444444) )        -
    DATA  (NAME (TESTU.DATA)      -
        FILE (VRDSDAT)            -
        UNIQUE                    -
        CYLINDER(1)               -
        VOLUMES (666666) )
/*
/&
```

Loading Records into a File

After you have defined a file, you can load records into it with a processing program of your own or with the REPRO command. This section discusses only the latter.

The REPRO command causes IDCAMS to retrieve records from a sequential tape or disk file, an indexed-sequential file, or a VSE/VSAM file and store them, in VSE/VSAM format, in a key-sequenced, entry-sequenced, or relative-record file.

When records are loaded into a key-sequenced file, they must have unlike keys and be in ascending key sequence. Index entries are created and loaded into the file's index as CIs and CAs are filled up. Free space is left and records are stored on particular volumes according to key ranges, as indicated in the file's definition in the catalog.

You can copy an entire file or only parts of it. You can specify that a certain range of records is to be copied by indicating the locations in the input file where copying is to start and stop. These locations can be indicated in several ways:

- By key for indexed-sequential or key-sequenced files (FROMKEY, TOKEY).
- By RBA for key-sequenced or entry-sequenced files (FROMADDRESS, TOADDRESS).
- By relative-record number for relative-record files (FROMNUMBER, TONUMBER).
- By number of records for any type of file (SKIP, COUNT).

The file into which records are copied may either be empty (that is, newly allocated by way of a DEFINE CLUSTER command), or may already contain records.

See “Reorganizing a File” on page 42 for information about what happens when records are added to an empty or nonempty key-sequenced, entry-sequenced, relative-record, or sequential file.

You can load all of the records in one job or in several jobs. In subsequent jobs, VSE/VSAM continues to store records as before, extending the file as required. For subsequent loads, KSDS input records must be arranged in key sequence, with *no* duplicate keys. If the input records are to replace records already existing in the output file, specify REPLACE on the REPRO command.

The job control information for loading a file is the same as for other types of file processing. Because your VSE/VSAM files are already cataloged when you start a REPRO job, the function of job control here is merely to name the required files and the volumes on which they are stored and to indicate any user catalog(s), so that VSE/VSAM can look at the definitions of the files.

EXAMPLE 17 shows the basic loading function of the REPRO command: taking the records of a sequential file on a disk and storing them in a newly defined VSE/VSAM file.

```
// JOB    EXAMPLE 17
// ASSGN  SYS002,cuu
// DLBL   SAMFILE,'SAM1'
// EXTENT SYS002,UVOL01,1,0,120,2
```

```
// DLBL   RRDS,'STOCKINV',,VSAM
// DLBL   IJSYSUC,'VSE.USER.CATALOG',,VSAM
// EXEC   IDCAMS,SIZE=AUTO
   REPRO  INFILE(SAMFILE,ENV(RECFM(F),BLKSZ(100))) -
          OUTFILE(RRDS)
/*
/&
```

Job control statement:

- // DLBL SAMFILE indicates the nonVSAM input file.
- // DLBL RRDS indicates the VSE/VSAM file that is to be loaded with records.
- // DLBL IJSYSUC indicates the job catalog in which the VSE/VSAM file (STOCKINV) is defined.

The REPRO command copies all the records from the input file SAM1 to the output file STOCKINV.

- The INFILE parameter points to the // DLBL statement that identifies the source, or input, file: SAM1.
- The OUTFILE parameter points to the // DLBL statement that identifies the file into which the input records are to be copied: STOCKINV.

Alternate Indexes

Apart from the prime index that VSE/VSAM automatically creates for every key-sequenced file, you can have VSE/VSAM build one or more alternate indexes over a single key-sequenced or entry-sequenced file. Every alternate index accesses the data records of a given file using a different key field (alternate key) within these records.

An alternate index, therefore, provides a unique way to gain access to the same base data, so that when you have several alternate indexes, you can access a file in several different ways without having to keep several copies of the same information organized differently for different applications. For example, a payroll file originally indexed on employee number can be indexed on additional fields (called alternate keys) such as employee name, department number, position code, skill code, or social security number.

The data over which the alternate index is built is referred to as the *base cluster*. It can be either a key-sequenced file or an entry-sequenced file, but not a relative-record file or a file which has been defined as reusable.

The alternate index itself is a key-sequenced cluster, namely the alternate-index cluster. It is usually referred to simply as the alternate index. Like an indexed base cluster, it consists of an index component and a data component. The index component of an alternate index is identical in structure, format, and function to the index component of a base cluster, that is, the prime index. The data component of an alternate index has a fixed format and serves to establish the relationship between the alternate index and the base cluster.

When building an alternate index, you can use as the alternate key any field in the base cluster's records which has a fixed length and a fixed position within every record. The alternate key must be in the first segment of a spanned record. For every alternate key value, the data component of the alternate index contains a unique record. This record consists of the alternate key itself, followed by a pointer that is the prime key (for a key-sequenced base) or RBA (for an entry-sequenced base) of the base-cluster record that contains the alternate key. If several base-cluster records contains the same alternate key, the alternate index record contains a pointer to every base-cluster record. These duplicate (or nonunique) keys are discussed under “Alternate Keys” on page 311.

Alternate-Index Path

To gain access to a base cluster using an alternate index, you must define a path between the alternate index and the base cluster. When you define a path with IDCAMS, you must specify the name of the alternate index which is to be considered as the entry for the path. The termination of the path is the base cluster to which the alternate index is related. A path requires a name of its own which always refers to the alternate index and the related base cluster as a pair. To use the alternate sequence to process records in the base cluster, you specify the CLUSTER name of the path in the 44-byte file ID of the // DLBL statement used to open the file.

When a path is referenced, both its alternate index and its base cluster are opened. If you want to process only the alternate index, without its base cluster, you can use the name of the alternate index instead of the path name and then process the alternate index like any other file.

Creating an Alternate Index

An alternate index can only be built over a nonempty VSE/VSAM base cluster. The logical steps involved in creating an alternate index are:

1. Define the alternate index and relate it to a base cluster by means of the DEFINE ALTERNATEINDEX command.
2. Build the alternate index either yourself or by means of the BLDINDEX command. The IDCAMS routines then perform the following operations:
   a. Extract the alternate key and the prime key or RBA from every record of the base cluster during a sequential scan through the base cluster.
   b. Order the extracted alternate keys, together with the associated pointers, by the alternate keys.
   c. Build the alternate-index records from the ordered key and associated pointers. (Records with the same alternate key are merged into a single alternate-index record.)
   d. Build the alternate index from the individual alternate-index records as a key-sequenced file.

Defining an Alternate Index

For the format of an alternate-index record, see Appendix B, “Format of an Alternate-Index Record,” on page 311.

When a DEFINE command is used to define an alternate index, VSE/VSAM creates three entries in the catalog: an entry for the alternate index, its data component, and its index component.

Attributes of the data and index components can be specified separately from the attributes of the alternate index as a whole. If attributes are specified for the alternate index as a whole and are not specified for the components, the attributes of the alternate index (except for its passwords and other protection attributes) apply to its components. If an attribute that is applicable to the data or index component is specified for both the alternate index and the component, the component specification overrides the alternate index specification.

You can also specify that the data and index components of an alternate index are to reside on different volumes. You do this by specifying the VOLUMES parameter as an attribute of both DATA and INDEX.

On ECKD devices, unique datasets (CLUSTER or AIX) must not allocate mixed extents of FAT-DASD and non FAT-DASD. The mix of Small and BIG-DASDs (3390 mod 9/27) (max. 10017 cylinders) is still allowed and not affected.

- In order to define a CLUSTER or AIX as UNIQUE with option FAT-DASD, make sure that each volume used for the assigned extents
  - either is predefined as FAT-DASD to the current VSAM catalog,
  - or has a minimum real capacity of 64K+1 tracks.

See “DEFINE ALTERNATEINDEX” on page 79 to identify which parameters can be used with every type of entry.

EXAMPLE 18 and EXAMPLE 19 show how to use the basic DEFINE parameters to define an alternate index. The examples do not consider all of the possible options. See also “Example 8: Creating an Alternate Index and Its Path” on page 223.

No // DLBL and // EXTENT statements are required if space for the alternate index is suballocated from one or more existing data spaces. VSE/VSAM selects which data spaces or portions of data space(s) on a volume to suballocate to the alternate index. You only indicate the volume(s) on which the alternate index is to be allocated and the amount of space to be allocated to it.

In EXAMPLE 18, the alternate index named DEPTIND is suballocated 100 tracks from data space(s) on volume VOLS0A. The alternate index is related to a base cluster named PAYROLL1. The alternate key starts at position 1 (relative to 0) in the data records of the base cluster and has a length of 12 bytes.

```
// JOB   EXAMPLE 18
// EXEC  IDCAMS,SIZE=AUTO
   DEFINE  ALTERNATEINDEX(NAME(DEPTIND) -
             RELATE(PAYROLL1/DEPT27R)   -
             VOLUMES(VOLS0A)            -
             TRACKS(100)                -
             RECORDSIZE(100 200)        -
             KEYS(12 1))                -
           CATALOG(VSAM.MASTER.CATALOG/UPDPW1)
/*
/&
```

EXAMPLE 19 defines an alternate index over a base cluster. The alternate index is defined as unique, so you must provide EXTENT information together with the definition of the alternate index. The FILE parameters link the file information, which is entered into the catalog, with the space information contained in the job control statements. By default, the alternate index is not reusable, has a key length of 64 beginning at byte 0, an average record size of 4086 bytes, and a maximum record size of 32,600 bytes.

```
// JOB     EXAMPLE 19
// DLBL    IJSYSUC,'VSE.USER.CATALOG',,VSAM
// DLBL    DFILE,,,VSAM
// EXTENT  ,UVOL0C,,,240,20
// DLBL    XFILE,,,VSAM
// EXTENT  ,UVOL0C,,,260,20
// EXEC    IDCAMS,SIZE=AUTO
   DEFINE  AIX                                  -
              (NAME(AIX.PAYROLL2.NAMES)         -
              RELATE(PAYROLL2)                  -
              RDPW(AIXRDPW)                     -
              VOL(UVOL0C)                       -
              UNIQUE)                           -
            DATA (NAME(AIX.PAYROLL2.NAMES.DATA) -
              CYLINDERS(1 1)                    -
              EXCEPTIONEXIT(ERREXIT1)           -
              FILE(DFILE))                      -
            INDEX (NAME(AIX.PAYROLL2.NAMES.INDEX) -
              CYLINDERS(1 1)                    -
                FILE(XFILE))
/*
/&
```

Building an Alternate Index

After you have defined an alternate index, you must build it from records in the base cluster (the base cluster cannot be empty) by using the BLDINDEX command.

BLDINDEX reads every base cluster record and forms a sort record consisting of the alternate key together with the prime key for a key-sequenced file or the RBA for an entry-sequenced file. These records are then sorted into alternate index key sequence. If the caller of BLDINDEX has provided enough GETVIS storage, these records are sorted internally.

You can determine the amount of GETVIS storage required to sort the records internally by using the following calculation:

1. Sort record length = alternate key length + prime key length (for a key-sequenced file) or alternate key length + 4 (for an entry-sequenced file).
2. Sort record length x number of records in the base cluster.
3. Sort area size = result of 2 above, rounded up to 32,768 or to the next multiple of 2,048, whichever is greater.
4. Sort table size = (sort area size ÷ sort record length) x 4.

The sum of 3 and 4 above is the required amount of GETVIS storage for an internal sort. This amount is in addition to the normal storage requirements for processing an IDCAMS command.

If BLDINDEX is unable to obtain enough GETVIS space to perform an internal sort, BLDINDEX dynamically defines two VSE/VSAM files and uses them as work files for an external sort. The sort work files are deleted at the end of the sort.

When EXTERNALSORT is specified, an external sort is performed only if not enough GETVIS space is available for an internal sort. If sufficient GETVIS space is available, BLDINDEX performs an internal sort.

The minimum amount of virtual storage required for an external sort is calculated as follows:

32,768 + (32,768 ÷ sort record length) x 4

The amount of space that IDCAMS requests when defining every sort work file is calculated as follows:

1. Sort records per block = 2041 ÷ sort record length
2. Primary space allocation in records = (number of records in base cluster ÷ sort records per block) + 10
3. Secondary space allocation in records = (primary space allocation x .10) + 10

Both primary and secondary space allocations are requested in records with a fixed-length record size of 2041 bytes and a CI size of 2048 bytes.

After the records have been sorted into alternate key sequence, BLDINDEX uses them to form the alternate index records. Every sort record is used to create one alternate index record, unless the NONUNIQUEKEY attribute has been specified in the definition of the alternate index. In the case of UNIQUEKEY, every alternate index record contains the alternate index key and the associated base cluster prime key (or RBA for an entry-sequenced file). In the case of NONUNIQUEKEY, every alternate index record contains one alternate key and all associated base cluster prime keys or RBAs representing records containing that same alternate key. If the alternate index has been defined with the REUSE attribute, BLDINDEX automatically writes the new alternate index records starting from the beginning of the file and overriding any records previously stored.

The parameters of the BLDINDEX command are used to identify the:

- Object over which the new alternate index is to be built (INDATASET),
- Alternate index itself (OUTDATASET),
- Sort work volumes (WORKVOLUMES), and
- Name of the catalog in which the sort work files are to be defined if they are required (CATALOG).

The EXTERNALSORT parameter indicates that an external sort is to be performed if insufficient storage exists for an internal sort.

EXAMPLE 20 shows the parameters.

```
// JOB   EXAMPLE 20
// EXEC  IDCAMS,SIZE=AUTO
   BLDINDEX  INDATASET(EXAMPLE.KSDS2)              -
             OUTDATASET(EXAMPLE.AIX/AIXUPPW)       -
             WORKVOLUMES(M3380A)                   -
             CATALOG(VSAM.MASTER.CATALOG/MCATMRPW) -
             EXTERNALSORT
/*
/&
```

Defining a Path

The DEFINE PATH command is used to establish the relationship, the *path*, between an alternate index and its base cluster. A path does not occupy any data space; it is a catalog entry only. The base cluster and its alternate index must already be defined when you define the path that relates them.

When your program opens a path for processing, both the alternate index and its base cluster are opened. When data in the base cluster is read or written using the path's alternate index, keyed processing is used; RBA processing is not allowed.

You can also establish a path directly over a base cluster, without an intermediary alternate index and with its own protection attributes. A path so defined provides access for a file under another name.

You can specify NOUPDATE access for the base cluster, which bypasses allocation of the base cluster's upgrade set and thus does not cause upgrading.

Specifying Information that Defines a Path

The examples given here are intended to demonstrate how to use the basic DEFINE parameters to define a path without considering all of the options. See also “Example 8: Creating an Alternate Index and Its Path” on page 223.

EXAMPLE 21 defines a path named PATH.PAYROLL1.NAMES over an alternate index named AIX.PAYROLL1.NAMES and its base cluster (named in the RELATE parameter when the alternate index was defined).

```
// JOB   EXAMPLE 21
// EXEC  IDCAMS,SIZE=AUTO
   DEFINE  PATH                          -
            (NAME(PATH.PAYROLL1.NAMES)     -
             PATHENTRY(AIX.PAYROLL1.NAMES) -
             MRPW(MASTER))                 -
           CATALOG(VSAM.MASTER.CATALOG/MRCATPW1)
/*
/&
```

Accessing a Base Cluster using a Path

A path is a means for accessing a base cluster either using an alternate index or, in the case of a base-cluster-only path, using another name. Like an alternate index or cluster, a path always requires a name of its own, which you specify in the DEFINE PATH command of IDCAMS. This name is also used as the "file ID" operand in the DLBL statement, which you use when you want to open the path. Opening a path results in associating the base cluster with the alternate index (for an alternate index path).

You may issue the same requests for an alternate-index path (GET, PUT, and so on) that you can issue against a VSE/VSAM file. However, all access to the base cluster using the alternate index by way of a path must be by key. Addressed access (that is, processing by RBA) and CI processing are not permitted. All key references are to the alternate-key sequence. If an alternate key occurs in several base cluster records, the base records are returned in the order in which they are stored in the file.

Alternate-Index Upgrade

All changes in the base cluster that affect the contents of its alternate index or indexes should be reflected in the base cluster's alternate index(es). It is to ensure that an alternate index is always "synchronized" with its base cluster.

This updating activity is referred to as *alternate-index upgrade*.

You may have VSE/VSAM upgrade an alternate index or you may upgrade it yourself.

Upgrade by VSE/VSAM

To have VSE/VSAM upgrade an alternate index, specify the UPGRADE attribute when you define the alternate index. As a result, that alternate index becomes a member of the upgrade set of the associated base cluster. Whenever you open the base cluster for any type of update processing other than CI access, VSE/VSAM opens all of the alternate indexes in the upgrade set and updates them, if necessary.

VSE/VSAM updates the affected alternate index(es) in the upgrade set of the base cluster whenever a base record is inserted or erased, or an alternate key field is changed. This updating activity is part of the request, and VSE/VSAM completes it before returning control to your program.

If the updating of an alternate index fails because of a logical error (such as a duplicate key condition for a UNIQUEKEY alternate index), the request which caused the update operation is rejected. At the same time, the base cluster (together with the alternate indexes of its upgrade set) is restored to the status existing before the request was issued (except for the sequence of the alternate index pointers).

If a physical error condition occurs, VSE/VSAM terminates the upgrading of the alternate index(es) immediately and enters the EXCEPTIONEXIT and/or SYNAD exit.

Upgrade by User

Because VSE/VSAM assumes that alternate indexes are synchronized with the base cluster at all times, You are responsible for upgrading the alternate index yourself:

- If you specify NOUPGRADE in the DEFINE ALTERNATEINDEX command, or
- If the base cluster is modified through CI access.

When you open a base cluster, those of its alternate indexes which have the NOUPGRADE attribute will not be updated by VSE/VSAM when you insert a record into, or erase a record from the base cluster, or change an alternate key field.

Defining a NonVSAM File

You use the DEFINE NONVSAM command to catalog any existing nonVSAM file into a VSE/VSAM catalog. An entry is created in a master or user catalog, but no space is allocated or reserved as a result of a DEFINE NONVSAM command.

When you define or delete a nonVSAM file in a password-protected catalog, the catalog's update password (or higher level password) is required.

EXAMPLE 22 shows how to define an existing nonVSAM file entry in the master catalog:

```
// JOB   EXAMPLE 22
// EXEC  IDCAMS,SIZE=AUTO
   DEFINE  NONVSAM              -
             (NAME(STOCKINV)    -
             VOLUMES(VOLS0B)    -
             DEVICETYPES(3380)) -
             CATALOG(VSAM.MASTER.CATALOG/UPDPW1)
/*
/&
```

See also “Example 4: Define NonVSAM and VSE/VSAM Files” on page 215.

Defining Work Files on Virtual Disk

Work files may reside on real disk devices or on *virtual disks*. For an outline on preparing for the use of virtual disk, refer to the "Work Files on Virtual Disk" in the VSE/VSAM User's Guide and Application Programming.

Figure 3 on page 38 is an example of how to define a user catalog, data space, and a cluster so to reside on virtual disk. In the example:

- EXPORT DISCONNECT is specified to remove a user catalog entry from the master catalog.
- DEDICATE is specified to reserve the whole disk for use by this catalog.
- SPEED is specified to reduce the number of disk accesses (that is, the space of the data component will not be preformatted).

```
// JOB VDISK DEFINE CATALOG,SPACE AND A CLUSTER ON VDISKS
// EXEC IDCAMS,SIZE=AUTO
 EXPORT USER.CATALOG.VIRTUAL DISCONNECT
 IF LASTCC=12 THEN SET MAXCC=0
 DEFINE USERCATALOG                  -
   (                                 -
     DEDICATE                        -
     NAME(USER.CATALOG.VIRTUAL)      -
     VOLUMES(FBA002)                 -
   )
 DEFINE SPACE                        -
   (                                 -
     DEDICATE                        -
     VOLUMES(FBA001 FBA003   )       -
   )                                 -
     CATALOG(USER.CATALOG.VIRTUAL)
 DEFINE CLUSTER                      -
   (                                 -
     BLOCKS(1000)                    -
     NAME(KSDS.CLUSTER.VIRTUAL)      -
     SPEED                           -
     VOLUMES(FBA001 FBA002)          -
   )                                 -
   DATA                              -
   (                                 -
     NAME(KSDS.CLUSTER.VIRTUAL.DATA) -
     KEYS(10 0)                      -
   )                                 -
   INDEX                             -
   (                                 -
     NAME(KSDS.CLUSTER.VIRTUAL.INDEX) -
   )                                 -
    CATALOG(USER.CATALOG.VIRTUAL)
/*
/&
```

Figure 3. Example: Defining Work Files on Virtual Disk

Altering Catalog Entries

Many of the attributes that you define when you create a catalog entry may be modified subsequently by the ALTER command.

Certain attributes of the file cannot be modified. For example, you cannot change the CI size and placement of the index in disk storage relative to the data of a key-sequenced file. Changing these attributes amounts to a reorganization of the file and requires that you define a new file and copy the old file into it.

Altering an object's entry normally does not require that the object's volume be mounted. This is because the object's use of space and the availability of space in the volume's data spaces can be determined by examining the catalog. However, the object's volume *must* be mounted whenever the volume's VTOC must be consulted or modified, such as when a data space, a unique component's space, or a catalog's space is to be altered.

-

Specifying Information That Alters an Entry

EXAMPLE 23 shows how to alter the CI and CA free space percentages of the data component of the file PAYROLL1. The ALTER of PAYROLL1.DATA shows a means of optimizing use of space for a file. The data component was originally defined with 40 percent free space. After initial loading this percentage is reduced, because further activity against this file will not be of the mass insert type.

```
// JOB   EXAMPLE 23
// EXEC  IDCAMS,SIZE=AUTO
   ALTER  PAYROLL1.DATA      -
            FREESPACE(10 10) -
          CATALOG(VSAM.MASTER.CATALOG/MRCATPW1)
/*
/&
```

EXAMPLE 24 shows a way to change the security scheme of a file by using the ALTER ACCOUNTS command.

Establishing a security scheme for an existing unprotected file would be done in the same manner.

```
// JOB   EXAMPLE 24
// EXEC  IDCAMS,SIZE=AUTO
   ALTER  ACCOUNTS/DEPT27MR -
            MRPW(DEPT26M)   -
            CTLPW(DEPT26C)  -
            UPDPW(DEPT26U)  -
            RDPW(DEPT26R)   -
            AUTH(D26AUTH)
/*
/&
```

See also "Example 7: Modifying and Listing the Cataloged Attributes of a File" on page 222.

Deleting Catalog Entries

You use the DELETE command to delete any previously defined VSE/VSAM object (data space, cluster, alternate index, path, catalog) and remove its catalog entry. In addition, you can remove the entry for a nonVSAM file from a catalog and optionally scratch that file from the volumes containing it.

All objects deleted by a single DELETE command must be defined in the same catalog.

Deleting a suballocated object normally does not require that its volume(s) be mounted, because its space allocation can be determined by examining the catalog.

If an object is deleted and the ERASE parameter was specified, then the space is not only freed for use by new objects but also overwritten with binary zeros. You can delete all alternate indexes and paths related to a base cluster by deleting the base cluster alone. However, even if you not only specified the ERASE parameter for the base cluster, but also specified the parameter during definition for all other objects connected to the base cluster, only the space freed by the base cluster will be overwritten with binary zeros. Note the following:

- To cause any related alternate indexes to be overwritten with binary zeros, you must delete them individually, before you delete the base cluster.
- When you delete entries from a user catalog, you may identify the catalog either with the CATALOG parameter, or (for the job catalog) the IJSYSUC DLBL job control statement.

  If the entries are to be erased, the catalog containing them is the catalog specified in the CATALOG *catname* parameter or the default catalog.

When a unique file or unique alternate index is deleted, the cluster or alternate index entry and the data and index entries are deleted from the catalog and the data space they occupied is also deleted. However, the volume entries will not be automatically deleted from the catalog. To delete the volume entries, you must specify SPACE in a separate DELETE command and all the data spaces on the volume must be empty.

A DELETE SPACE will cause all empty data spaces of the specified volume to be deleted rather than an individual data space. However, if FORCE is also specified, nonempty data spaces are deleted as well.

When a nonVSAM entry is deleted, the file entry is scratched from the VTOC of the pack on which the file resides unless the user specifies the NOSCRATCH parameter.

Specifying Information that Deletes an Entry

Example 25 deletes all of the objects defined in the user catalog and then deletes the catalog itself. The catalog cannot be deleted as long as it contains any entries (besides its own and the entry for its own data space).

```
// JOB   EXAMPLE 25
// DLBL  IJSYSUC,'VSE.USER.CATALOG',,VSAM
// EXEC  IDCAMS,SIZE=AUTO
   DELETE  PATH.PAYROLL2.NAMES   -
```

```
            PATH
      DELETE  AIX.PAYROLL2.NAMES    -
              ERASE                 -
              AIX
      DELETE  PAYROLL2              -
              PURGE                 -
              CLUSTER
      DELETE  VSAM.COMPRESS.CONTROL -
              CLUSTER
      DELETE  (UVOL0B)              -
              SPACE
      DELETE  VSE.USER.CATALOG-
              USERCATALOG
   /*
   /&
```

Example 26 shows the deletion of a base cluster, alternate index, and path. The alternate index and path are deleted implicitly as a result of deleting the base cluster.

```
   // JOB   EXAMPLE 26
   // EXEC  IDCAMS,SIZE=AUTO
      DELETE  PAYROLL -
              PURGE   -
              CLUSTER -
              CATALOG(VSAM.MASTER.CATALOG/MRCATPW1)
   /*
   /&
```

Example 27 shows the deletion, with the ERASE option, of a unique cluster.

```
   // JOB   EXAMPLE 27
   // EXEC  IDCAMS,SIZE=AUTO
      DELETE  ACCOUNTS -
              ERASE    -
              PURGE    -
              CATALOG(VSAM.MASTER.CATALOG/MRCATPW1)
   /*
   /&
```

See also “Example 18: Deleting Entries in a User Catalog and the User Catalog Itself” on page 233 and “Example 24: Deleting Entries in the Master Catalog and the Master Catalog Itself” on page 237.

Using REPRO for Catalog Backup and File Reorganization

Note: REPRO *cannot* be used for reorganizing catalogs. The only way to reorganize a catalog is by creating a new catalog and restoring all data sets to the new catalog.

Backing Up a Catalog

The REPRO command lets you recover a catalog that has become inaccessible. You create a backup copy (with REPRO) and reload it again (with REPRO). Do not use this function to make the catalog bigger. You should unload your catalog periodically (with REPRO) to always have a recent backup copy available.

Unloading a Catalog

You can use the REPRO command to unload a catalog to a sequential (SAM) file, or to a key-sequenced or entry-sequenced file. (You must unload a master catalog to a sequential file, or else the subsequent reload will not work.) Do not allow the catalog to be updated during the unload operation.

If you use a key-sequenced or entry sequenced file to provide backup for a user catalog, define the file that contains the unloaded catalog in *another* catalog. This allows you to recover the unloaded catalog if the original (copied) catalog is destroyed.

Use LISTCAT before unloading a catalog so that you can compare the listing of the unloaded catalog with the listing you obtain after reloading.

Reloading a Catalog

You can use REPRO to reload the backup copy of a catalog into a catalog (the "target") with the same name, volume number, and device type as the original catalog.

The target catalog can be an earlier or a later version of the original catalog, or a newly-defined user catalog (without any entries other than the basic catalog information). The newly-defined catalog must have the same extents as the original catalog had at the time the backup copy was made. If you need to enlarge the catalog, use EXPORT and IMPORT.

If you define a bigger catalog and reload the old catalog into it with REPRO, it may be destroyed when next processed.

Reloading a version of the original catalog results in a catalog equivalent to the original one at the time the backup was made. Reloading:

- Replaces entries in the target catalog with entries (of the same name) from the backup.
- Inserts, into the target catalog, entries that exist only in the backup.
- Deletes, from the target catalog, entries that exist only in the target catalog.

During reloading, up to 100 messages may be issued to indicate entries that exist only in the target catalog or only in the backup. Reloading a newly defined catalog has the same results as reloading a version of the original catalog, with one exception. The newly-defined catalog's volume record contains only self-describing information. (The assumption is that, if a catalog is new, no other VSE/VSAM data space exists on the volume under normal conditions.) If the catalog is new, the reload operation bypasses the reload of the original version of the volume record; all the data space information previously in the volume record at the time the catalog was unloaded is lost.

After the reload of a newly-defined catalog, the entries in the reloaded catalog may reference files previously existing on the volume.

If these files still exist, they can be accessed, but any attempt to write to the data space in which they reside may fail. In this situation, you can restore the needed information to the volume record as follows:

1. Use EXPORT PERMANENT to remove the file entries from the new catalog.
2. Define a data space large enough to accommodate the files.
3. Use IMPORT to put the files into the newly defined data space.

The space allocation used for the reloaded catalog is the space allocation of the target catalog whether it is a newly defined catalog, or an earlier or later version of the unloaded catalog. In both cases, the catalog allocation properly reflects the VSE/VSAM data space for the catalog on the volume.

A DLBL job control statement (IJSYSUC) should be used to identify the catalog to be unloaded or reloaded. This ensures that the catalog can be opened as a catalog prior to an unload or a reload operation. The define of a new catalog and the listing of a catalog should be done in separate steps from any reload operations.

Password protection of VSE/VSAM catalogs is optional. You can establish passwords at the time of catalog definition, or you can add them subsequently. The password can be supplied in the REPRO command or through the system console in response to password prompting. When a catalog is password protected, the master password must be provided to permit catalog unloading and catalog reloading.

Notes and Restrictions

The following restrictions should be observed when a catalog backup operation is to be done:

- Unload the master catalog only to a sequential file.
- The unloaded version of a catalog must be reloaded into a catalog with the same name, volume number, and device type.
- The unloaded version of a catalog must be reloaded into a catalog with the same entry capacity (that is, number of entries cataloged). Some high key range compression may occur when reloading into a newly defined catalog; consequently, the reloaded version may require less high key range space than

a source catalog with several extents. The target catalog will not be extended by means of secondary allocations during the reload operation.

- After you reload a catalog, use LISTCAT to list its contents. Run LISTCAT in a separate job step, so that the catalog will be closed after it is reloaded (to update its self-defining information). Compare the listing with the one you obtained before unloading the original catalog to ensure that you have used the right backup.
- When reloading or restoring a catalog, a LISTCAT command should be executed with every unload and after every reload operation (in a separate step after a reload). The LISTCAT output obtained in conjunction with unload may be necessary for comparison to the LISTCAT output obtained after reload. Both outputs may be necessary to determine the manual intervention required when mismatches are detected during reload.
- Files or volumes (in the original catalog) that are deleted and redefined or permanently exported after an unload operation will not be flagged as changed upon reload because detection of this type of change is not possible. The deleted files or data spaces will still be defined in the restored catalog. (Entries that exist only in the backup copy are inserted into the target catalog.) Any attempt to process these entries will yield unpredictable results because the space reflected in the catalog may no longer be owned by the catalog. The catalog may be corrected by reissuing the DELETE commands.
- If VSE/VSAM file or data spaces have been defined or imported since the last catalog backup and the catalog is reloaded or restored, then the defined files or data spaces will not be defined in the reloaded or restored catalog. Processing these files or data spaces by means of the restored catalog is not possible, because they cannot be accessed. The space formerly occupied by these VSE/VSAM files or data spaces will not be usable, but may be recovered by scratching the format-1 labels in the VTOC for the data spaces. If any volumes were added to the catalog (between the backup and the recovery), they will also be unusable until you use the DELETE command with FORCE option to give up volume ownership.
- If a VSE/VSAM file has been extended since the last catalog backup, the new extents will not be defined in the restored or reloaded catalog. Any attempt to process records in the added extents will result in a logic error. If the file has been extended within space already allocated to the file before the backup but has acquired no new extents, then you can issue the VERIFY command to update the catalog pointers, and the file may be accessed normally.
- The data in any extents that have been acquired by the file since the catalog was backed up is not recoverable. For an entry-sequenced file the data in any new extents should consist only of records that have been added to the end of the file. Therefore, it is possible to recover all of the data in the old extents by accessing the file sequentially up to the end of the old physical space allocation. For a key-sequenced file, the new extents may be any portion of the file because of control-area splits. An attempt to read the data in logical sequence will fail with an invalid RBA indication when the data in the new extents is reached. You could access the key-sequenced file by means of address sequence, but you then have the problem of identifying the missing records. Individual file recovery for those files affected will be necessary.

  For a discussion of making the contents of the backup catalog agree with the contents of the original catalog at the time it became inaccessible, refer to the "Procedures for VSE/VSAM Recovery" in the VSE/VSAM User's Guide and Application Programming.

Reorganizing a File

(REPRO *cannot* be used for reorganizing catalogs. The following explanations do not apply to reorganizing a catalog.)

You use the REPRO command to reorganize an old file by copying it into a newly-defined file of the same type. With key-sequenced files, for example, you can specify different percentages of distributed free space and different performance options for the new file when you define it.

Because data is copied as single logical records, automatic reorganization takes place when you copy a key-sequenced file into another key-sequenced file. Reorganization includes:

- Relocation of records so that their entry sequence matches their key sequence.

- Redistribution of the free space throughout the file.
- Reconstruction of the prime index.

Fixed-length, unblocked records in an indexed-sequential file are preceded by the key string when they are written into a VSE/VSAM file. Therefore, the length of the records in the output file must include the length of the key string.

The new (output) file into which records of the old (input) file are copied may either be empty or already contain records. Table 4 on page 43 shows, by example, what happens when records from the input file are added to an empty or nonempty file.

Table 4. Example: Empty Versus Nonempty Output File

Input File	Empty SEQ	Output KSDS	File RRDS	Nonempty SEQ	Output KSDS	File RRDS
SEQ*	1	2	5	3	4	7
KSDS	1	2	5	3	4	7
RRDS, VRDS	1	2	6	3	4	8

Note: * SEQ = VSE/VSAM ESDS or SAM file

1 = Copies in sequential order.

2 = Copies ordered by defined key field and builds an index. The source records must be in key sequence.

3 = Adds records in sequential order to the end of file (VSE/VSAM ESDS); copies records in sequential order from the beginning of the file overlaying previous data (SAM sequential). This action will occur only if the operator responds with DELETE when the VSE message OVERLAP ON UNEXPRD FILE or DUPLICATE FILE ID is encountered.

4 = Merges records by the defined key field and updates the index. A record whose key duplicates a key in the output file is lost, unless you specify that the new record is to replace the old one.

5 = Copies records sequentially into consecutive slots, beginning with 1.

6 = Copies records in the same position they had in the input file.

7 = Does not copy a record unless the file has the REUSE attribute and the REPRO REUSE option is specified.

8 = Same as 6 except that a record in the input file that has the same number as one in the output file is lost, unless you specify that the new record is to replace the old one.

If you use REPRO to locate errors in a file, IDCAMS accepts a limited number of errors (as defined in the ERRORMAX parameter of REPRO) while trying to read the file and terminates the copy operation when this limit is exceeded. These so-called "acceptable" errors counting toward ERRORMAX are duplicate keys, wrong length records, records out of sequence, and I/O errors in the data component of a VSE/VSAM file.

You can save time in reorganizing VSE/VSAM files (using the REPRO command) by specifying the optional BUFSP or BUFNI and BUFND parameters in the DLBL statement. Ordinarily, REPRO uses default values for buffers that are based on the partition size and on available storage. By specifying an appropriate value in the BUFSP, BUFNI, or BUFND parameter, you can override the default; you can allocate additional data CI buffers for each of your VSE/VSAM files (if you have available virtual storage). For example, if your input file is key-sequenced with a 512 byte index CI and a 2048 byte data CI, you can specify BUFSP=10752 ((5 x 2048) + 512 = 10752) and cause five data CI buffers (and one index CI buffer) to be allocated when the file is opened.

For other uses of REPRO, see “Loading Records into a File” on page 31. See also “Example 5: Loading and Printing of Files” on page 218 and “Example 6: Modifying and Printing the Contents of VSE/VSAM Files” on page 221.

Using EXPORT/IMPORT for Transporting or Backing Up Files

You use the EXPORT and IMPORT commands to transport VSE/VSAM files and user catalogs between z/VSE systems (or set of systems in a disk sharing environment) or between z/VSE and z/OS systems. The EXPORT command extracts catalog information and produces a portable copy of the file that is to be moved. The IMPORT command loads a portable file and its catalog information in the receiving system. Figure 4 on page 44 compares volume and file portability. File portability is achieved by moving volumes or by moving individual files.

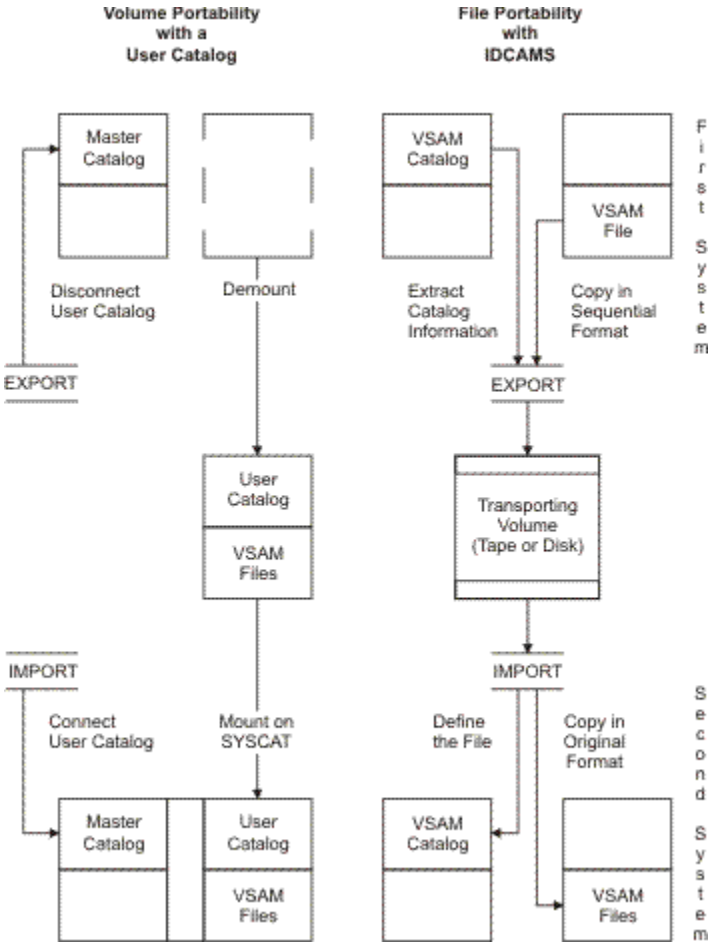


Figure 4. Data Portability by Moving Volumes or Files

EXPORT and IMPORT also enable you to create a backup copy of a file and its catalog entries and reload them into the same system when they are needed. When you import a backup copy, the catalog entries are regenerated.

For information on how to use the EXPORT/IMPORT commands to create backup copies of several VSE/VSAM files on a tape volume (or volume set), see “Multifile Volume Considerations for IDCAMS” on page 13.

Master and user catalogs cannot be copied using EXPORT and IMPORT; a user catalog can, however, be imported to another system by physically transporting the volume on which it is stored. When a user catalog is to be transported, it is not copied; the user catalog remains on its original volume in its original form. When it is exported by an EXPORT DISCONNECT command, the master catalog's pointer to it is removed. When it is subsequently imported to a new system by an IMPORT CONNECT command, a pointer to it is created in the new system's master catalog. (VSE disk sharing supports shared master catalogs.) You do not have to issue the EXPORT DISCONNECT command to move a user catalog to another system, but must use an IMPORT CONNECT command to build a pointer to the user catalog in the new system's master catalog. If you did not issue the EXPORT DISCONNECT command for it, then it

belongs to both systems. If both systems support VSE disk sharing and the user catalog is on a shared volume, it can be shared. Otherwise, the user catalog can be available to only one system at a time.

When a file is exported, relevant portions of its catalog entries are copied to a movable volume along with the cluster's component(s). The catalog name cannot be moved with the file. The portable copy is a variable-blocked, spanned, sequential file; it may be stored on tape or disk. You can control the block size of the portable file. If the file is exported to tape, if it is to be other than the first file on the tape and if the tape has just been mounted, you must issue the MTC command to correctly position the tape. Exportation of a file is either permanent or temporary. In permanent exportation, the catalog entries are deleted and storage space is freed in the original system. In temporary exportation, the sending system retains a copy of the file, but the copy is marked to indicate that there is a copy elsewhere.

When a base cluster and its alternate index(es) are permanently exported, the alternate index(es) must be permanently exported before the base cluster. Otherwise, the alternate index(es) will be deleted when the base cluster is deleted and cannot be exported. EXPORT will automatically export paths related to the cluster or alternate index being exported. When an object is exported, the statistics kept in its catalog entry are not extracted for exportation and are not available when the object is subsequently imported.

When a base cluster and its alternate index(es) are imported, the base cluster must be imported before the alternate index(es). If you are using a tape as the transporting volume, you must first forward space the tape to the base cluster and read it, then rewind to the load point and read the alternate indexes sequentially. For more information, see “Multifile Volume Considerations for IDCAMS” on page 13. A path cannot be transported as an object from one system to another, but is automatically imported when its related base cluster and alternate index are imported.

You may alter some of a file's attributes when you import it, but space allocation and buffer space problems may occur if a VSE/VSAM file is imported to a device of a type different from the type it was exported from. The space allocation quantities are recorded in terms of tracks (minimum CAs) (even if you specified CYLINDERS, BLOCKS, or RECORDS in the DEFINE command) in a file's catalog entry. When the file is imported the number of minimum CAs in the catalog entry is not changed to reflect the characteristics of a new device type unless you import the file into a predefined file with adjusted allocation quantities.

For example, an attempt to export a VSE/VSAM file from a 3380 and import it to a 3375 may fail because the allocation quantities in the catalog entry specify less space on a 3375 than they did on a 3380. (If the secondary space allocation quantity is not zero, VSE/VSAM may be able to allocate enough secondary space on the IBM 3375 to contain the file.) Conversely, if a file is exported from a 3375 and imported to an IBM 3380, it may be allocated more space than it needs.

You may use EXPORT and IMPORT to redefine a file for a different device with space parameters that are appropriate to the new device. You may also change protection attributes. To change the file's device type, you permanently export it (or temporarily export and delete it), redefine it (by way of DEFINE) with new allocation attributes, then import the transporting copy into the newly defined file. (If you use DEFINE before deleting the exported file, you can use the exported file's entry as a model assuming you give it a new name.) The newly defined file must meet the following conditions:

- It must be empty.
- It must be the same type of file (INDEXED, NONINDEXED, or NUMBERED) as the file being imported.
- If INDEXED, it must have the same key length and key position as the file being imported.
- It must have a maximum logical record length greater than or equal to that of the exported file.

To change devices without allocation problems when moving clusters or alternate indexes between systems or between catalogs, you may use two techniques.

The first technique is:

1. Use the EXPORT command to create the copy of the cluster or alternate index to be exported.
2. Use the DEFINE command to define a new entry for the cluster or alternate index in the catalog to which the cluster or alternate index is to be imported. Specify all the parameters used when the cluster or alternate index was originally defined, using the MODEL parameter if a catalog that contains a suitable model entry is available. If space was allocated in RECORDS, you may specify the same

quantity; if it was allocated in TRACKS or CYLINDERS, you must adjust the quantity for the new device type. If an entry already exists in the catalog for this file, you must delete that entry or use a different name in the DEFINE command and specify the new name in the OBJECTS parameter of IMPORT.

3. Use the IMPORT command to read the cluster or alternate index into the predefined empty cluster or alternate index. If the empty cluster or alternate index was defined with a name different from that of the cluster or alternate index exported, the NEWNAME subparameter of the OBJECTS parameter must be used to rename the exported cluster or alternate index file ID.

The second technique is:

1. Use the REPRO command to create the copy of the cluster or alternate index to be exported.
2. Follow step 2 above except do not specify a new name by means of IMPORT.
3. Load the cluster or alternate index into the predefined cluster or alternate index by issuing a REPRO command to copy it.

EXPORT: Making a File Portable

The EXPORT command enables you to copy a cluster or an alternate index in sequential form to a storage volume to be transported to another system. The transporting volume may be magnetic tape or disk. In addition, the EXPORT command extracts information from the catalog entry that defines the object to be transported and copies the information to the transporting volume. The information is used to define the file automatically in a VSE/VSAM catalog in the receiving system.

VSE/VSAM always defaults to an optimum buffer space value when using the EXPORT command; it ignores the buffer allocation values in the catalog entry and in the DLBL statement. See the following examples:

- “Example 10: Exporting a VSE/VSAM File” on page 227
- “Example 11: Exporting an Alternate Index and Base Cluster” on page 228
- “Example 12: Disconnecting a User Catalog from a Master Catalog” on page 229

IMPORT: Loading a Portable File

The IMPORT command enables you to use the catalog information extracted by EXPORT to automatically define a new cluster or alternate index in the catalog that you specify and to sub-allocate space for it. The object itself is stored, in its VSE/VSAM format, in the space allocated for it. Alternately, you can use DEFINE to define a new cluster or alternate index and allocate space for the object and use IMPORT to copy the object from the transporting volume into the space allocated for it, that is, import the object into an empty cluster.

If you want the imported file to be in compressed format, you need to define the cluster with the COMPRESSED attribute prior to issuing the IMPORT command.

You can also use IMPORT to define a pointer to a user catalog in the master catalog. The user catalog is not copied, but remains on its original volume in its original form.

A cluster, alternate index, or user catalog cannot be transported to a system if its name or the name of any of its components already exists in the receiving catalog. The only exceptions are when the cluster or alternate index name that already exists in the receiving catalog belongs to an object that has been temporarily exported (that is, TEMPORARY was coded when it was exported), or belongs to the predefined (empty) object that is to receive the imported object. The entry of a temporarily exported object is deleted when an object with the same name is imported; a new entry is then built for the imported object. When a base cluster and its alternate indexes are imported, the base cluster must be imported first, followed by the alternate indexes in any order. This is necessary because a base cluster must exist before an alternate index can be defined over it.

You may specify any block size that you choose for the portable file at the time of its creation by EXPORT by using the BLOCKSIZE sub-parameter of the ENVIRONMENT parameter. The default value is 2048 bytes per block.

When an IMPORT is issued, you must specify the same BLOCKSIZE that you specified during the EXPORT operation. If no value is given, the system assumes a value of 2048 bytes per block.

VSE/VSAM always defaults to an optimum buffer space value when using the IMPORT command; it ignores the buffer space value in the catalog entry and in the DLBL statement. See the following examples:

- "Example 13: Importing a Base Cluster and Alternate Index" on page 229
- "Example 14: Importing an Entry-Sequenced File" on page 230
- "Example 15: Connecting a User Catalog to the Master Catalog" on page 231
- "Example 16: Using REPRO to Unload a User Catalog to Tape" on page 232

Listing Catalog Entries

You use the LISTCAT command to list entries from a given catalog. The listing shows information about objects defined in the catalog, such as:

- Attributes of the object.
- Creation and expiration dates.
- Protection specification.

  Passwords and other protection information in an entry are not listed unless you specify the master password for the file defined by the entry or the master password for the catalog itself.
- Statistics. That is, information regarding the dynamic use or accessing of the data represented by the entry.
- Space specifications and allocations.
- Volume information.

The ENTRIES (entryname) parameter is used to specify the names of individual entries to be listed. The parameters CLUSTER, DATA, INDEX, SPACE, NONVSAM, PATH, USERCATALOG, and ALTERNATEINDEX are used to specify types of entries to be listed. The parameters ALL, NAME, VOLUME, and ALLOCATION specify the fields to be listed for every catalog entry.

The combination of these parameters allows you to tailor the scope of the listing to meet your needs. See the chart in the LISTCAT command section to see the results of coding various combinations of entryname and type of entry (CLUSTER, DATA, etc.). See the following examples:

- "Example 3: Define VSE/VSAM Files" on page 212
- "Example 4: Define NonVSAM and VSE/VSAM Files" on page 215
- "Example 7: Modifying and Listing the Cataloged Attributes of a File" on page 222
- Appendix A, "Interpreting LISTCAT Output," on page 271

Printing Data Records

You use the PRINT command to list some or all of the records of a sequential, indexed-sequential, or VSE/VSAM file in one of the following formats:

- Every byte as two hexadecimal digits (HEX)
- Every byte as a single character (CHARACTER)
- A combination of these two, side by side (DUMP)

You may specify a range of records to be printed in the same way as you do for copying records, namely:

- By key for indexed-sequential or key-sequenced files (FROMKEY, TOKEY).
- By RBA for key-sequenced or entry-sequenced files (FROMADDRESS, TOADDRESS).
- By relative-record number for relative-record files (FROMNUMBER, TONUMBER).
- By number of records for any type of file (COUNT, SKIP).

The components of a key-sequenced file can be listed individually. To list a component of a key-sequenced file, specify the component name as the file ID in the DLBL statement. To print a user catalog as a key-sequenced file or to print any file cataloged in a user catalog, the user catalog must either be a job catalog or be specified in the CAT parameter of the DLBL job control statement.

Sequential, entry-sequenced, and relative-record files are listed in physical sequential order. Indexed-sequential and key-sequenced files can be listed in key order or in physical sequential order. A base cluster can be listed in alternate key sequence using a path.

Only the data content of records is listed. System-defined control fields are not listed. Every record listed is identified by one of the following:

- Its relative byte address (RBA) for entry-sequenced files.
- Its key for indexed-sequential and key-sequenced files.
- Its sequential record number for sequential and relative-record files.

If you use PRINT to locate errors in a file, IDCAMS accepts a limited number of errors (currently 4) while trying to read the file and terminates the operation when this limit is exceeded. These so-called "acceptable" errors are duplicate keys, wrong length records, records out of sequence, and I/O errors in the data component of a VSE/VSAM file. See the following examples:

- “Example 5: Loading and Printing of Files” on page 218
- “Example 6: Modifying and Printing the Contents of VSE/VSAM Files” on page 221
- “Example 8: Creating an Alternate Index and Its Path” on page 223
- “Examples of PRINT Command Output” on page 177

Verifying a File's Accessibility

The VERIFY command was formerly used to validate data in files that were not closed successfully because of a system failure or a user error. Job streams that still contain the VERIFY command will continue to run.

OPEN for output automatically verifies the catalog records of all files that were not properly closed, provided verification is required and possible during OPEN.

An OPEN for input does not correct the catalog records, so that a message with error code 118 is issued. To get rid of it, open the file for output.

In the following cases, VSE/VSAM does also *not* perform automatic verification; a message will tell you when to issue VERIFY:

- If you loaded a file with the SPEED option specified (see DEFINE CLUSTER) and a system failure occurs during the load, VERIFY cannot help because the file was not preformatted. You must reload the file.

  If the file is being extended (due to resume loading, adding records, and so on), VSE/VSAM always extends in RECOVERY mode. Thus, if the file is not properly closed, VSE/VSAM performs automatic verification the next time the file is opened.
- VSE/VSAM does not perform automatic verification on a SAM ESDS file defined with NOCIFORMAT. Issuing VERIFY is also not useful for this type of file.

  For a CI-format SAM ESDS file, the EOF indicator in the catalog is not updated, because the file is always loaded and extended in SPEED mode.
- VSE/VSAM does not perform automatic verification on an ESDS file opened for CNV access. However, in this case you can quite likely correct the file by issuing the VERIFY command.

Using BACKUP and RESTORE

Functions

With this function you can:

- Back up objects to tape or disk devices. Restore objects to device types that are different from the device types on which the objects originally resided. You can move objects freely between CKD and FBA devices.
- Change the allocation size for the data component of an object at restoration with the DATARECORDS parameter.
- Change the index CI size at restoration with the INDEXCISIZE parameter.
- Back up and restore empty objects.

Data is the most critical resource in data processing and should therefore be backed up regularly. This allows you to restore the latest backup copy in case the data is damaged or destroyed.

From the backup copy, the RESTORE command recreates a VSE/VSAM object that may have a physical structure different from that of the original object, but that behaves, from the user's point of view, exactly like the original object.

VSE/VSAM Objects Supported

These objects and their catalog information can be backed up:

- Key-sequenced data sets (KSDS)
- Entry-sequenced data sets (ESDS)
- Relative-record data sets (RRDS)
- Variable-length relative record data set (VRDS)
- Alternate indexes (AIX)
- SAM ESDS files in CI format
- Paths
- ESS volumes (in conjunction with the IDCAMS SNAP and IMPORT CONNECT commands)

An empty object is an object that:

- Was defined with the NOALLOCATION parameter, or
- Has never been loaded with data, or
- Has not been loaded since being reset.

Note: The file you are backing up must be available for an INPUT OPEN. Depending on the specification in the SHAREOPTIONS parameter, the OPEN *might* fail if the file is currently opened for input or output by another partition or system. Because the OPEN not necessarily fails, it is strongly recommended that the file which is being backed up should not be opened for output by any other partition or system. Otherwise, the resulting backup copy might not represent the actual state of the original file.

To enable a backup from target volumes after exploiting the FlashCopy® function of the Enterprise Storage Server® (ESS), the IDCAMS SNAP and IMPORT CONNECT commands must be used prior to the BACKUP command, in each case with a synonym list. As a result, the BACKUP command will use the target volumes created by SNAP and rendered accessible by IMPORT. For details, see “SNAP” on page 199, “IMPORT” on page 160, and “BACKUP” on page 72. See also "Using FlashCopy to Back Up VSE/VSAM Data Sets" in z/VSE Administration, which describes the dialog that supports this process and illustrates a sample job stream that is generated based on the dialog.

Note: Do not use BACKUP and RESTORE to transfer data between a SCSI device and a standard FBA or CKD device. Use EXPORT/IMPORT instead. Use REPRO for migrating to SCSI devices from clusters to which the IMBED, REPLICATE, or RECOVERABLE attributes currently apply. These attributes are no longer supported.

Device Types Supported for Backup Files

The *backup file* (also referred to as backup copy) can be created on magnetic tape or disk volumes.

For *backup to tape*, you can use any of the tape devices that are supported by z/VSE. Both start/stop and streaming mode are supported, but streaming mode is ensured only in single-batch environments with certain buffer sizes. Streaming mode is not ensured when file modifications (that is, moving files to a volume of a different device type, or specifying new data component allocation sizes or index CI size) are made at restoration.

Backup to tape also permits saving objects from more than one catalog on the same tape, by executing the backup procedure multiple times in succession in a single job.

For *backup to disk*, you can use any of the disk devices that are supported by z/VSE.

Target Volume for Restoration

The objects backed up can be restored to the same or to a new set of volumes. You can migrate objects freely between CKD and FBA disk devices. The only restriction in moving files to a different device is that the data CI size must not be changed in the transition. This is necessary to avoid record-level reorganization with its undesirable performance characteristics.

During migration to a device of different type, allocation sizes may change to accommodate the new device characteristics.

When restoring to the original volume, the location of suballocated files on the volume is mainly controlled by VSE/VSAM. For unique files, you must specify where you wish to files to go.

Backing Up or Restoring Several VSE/VSAM Objects

You can back up or restore up to 255 VSE/VSAM objects with a single command if you list them.

Alternate indexes and paths are automatically backed up with their related base clusters or path entry clusters and thus do not reduce the number of objects that can be specified.

Using Generic Names

It may often be desirable to back up or restore several related VSE/VSAM objects under a generic name, that is, the initial characters common to the names of all objects wanted. A generic name usually ends with an asterisk (*). For example, ORD* identifies all items whose names begin with the characters ORD. Thus you can back up all VSE/VSAM objects of a catalog or a desired subset of objects or you can restore all objects of a backup file or a subset of them.

Excluding Objects from Backup/Restore

The generic name you want to use for backup or restoration may comprise some object(s) you do not wish to back up or restore. Such objects can be excluded from backup or restoration by specifying them (in the form of an entryname or another generic name) in the EXCLUDE parameter of the BACKUP or RESTORE command.

The EXCLUDE parameter can also be used to suppress automatic backup/restoration of alternate indexes, path entries, and empty objects.

VSE/VSAM compression control data sets will always be excluded, because the compression-related information is backed up with each compressed cluster.

Reorganizing Control Areas

Restoration of a KSDS causes the reorganization of the data component on the basis of control areas (CAs). Control areas that were not physically adjacent (that is, CA splits) are physically adjacent after restoration. This avoids unnecessary disk arm movements for sequential processing. (Note that the RBAs of the CAs will change accordingly.)

Backup/Restore can therefore be used to reorganize KSDSs on the basis of CAs.

Changing a File's Allocation Size at Restoration

You can change the amount of space allocated to a file at restoration by the DATARECORDS parameter. This is particularly useful if you restore objects to a device type other than where they were backed up from. You can specify device-independent values and the program chooses values appropriate to the new device type.

Catalog Migration

You cannot back up and restore catalogs, but when you back up and restore objects (including empty objects), their catalog information and compression control information is backed up and restored too.

This makes it possible for you to use BACKUP and RESTORE to copy objects and their catalog information into a different catalog. If the new catalog already contains an entry with the same name as the object being restored, that entry is overwritten with the new information.

Advantages of BACKUP and RESTORE over EXPORT and IMPORT

Reduced Specification Effort

Several objects can be backed up or restored with a single command. This is even easier with the use of generic names and the EXCLUDE parameter.

Generic backup and restoration of empty objects means you no longer have to recreate empty objects when restoring objects into a catalog that has been destroyed.

Alternate indexes and paths are automatically backed up or restored. This automatic backup or restoration can be suppressed if desired.

Backup Cross-Reference Listings

Because the backup file may be on several tape or disk devices, it may not be apparent how the objects are distributed over the volumes. Therefore, at the end of processing of every BACKUP command and of RESTORE command if XREF or XREFONLY parameters are specified, VSE/VSAM Backup/Restore prints *cross-reference listings* of all objects backed up and their place on the tape or disk devices.

The Backup cross-reference listings also show all empty objects.

For a backup file on tape, two listings are provided:

- *Volume cross-reference listing*. The entries are sorted by volume sequence number.
- *Object cross-reference listing*. The entries are sorted by object name.

For a backup on disk, three listings are provided:

- *Extent cross-reference listing*. The entries are sorted by extent sequence number.
- *Object cross-reference listing*. The entries are sorted by object name.
- *Extent list*. The entries contain descriptions of all disk extents used by the backup file. For each disk extent, the list shows a) the volume serial number of the volume where the extent is located, and b) the limits of the extent. Also, the list shows the disk address of the backup file record that was written last; from this information, the user can see how far the last disk extent is filled up with data.

For examples, refer to “Backup Cross-Reference Listings” on page 240. For a description of the RESTORE command, refer to “RESTORE” on page 189.

Easier Handling of Backup Volumes

If permanently resident disk volumes are used for backing up and restoring VSE/VSAM objects, operator intervention is not required during the backup and restoration processes.

You can restore a backed up file from several volumes selectively:

- If the backup file resides on disk volumes, you need only make available those volumes or disk extents that contain the data to be restored. You do this by providing the appropriate // EXTENT statements in the restore job.
- If the backup file resides on tapes, you need only mount those tapes that contain the data to be restored. Which of the volumes need to be made available through JCL statements or must be mounted can be determined from the backup cross-reference lists produced after the execution of the BACKUP command.

  Note that if you have assigned an alternate tape drive, processing automatically switches to the next backup volume (on the alternate tape drive) when the current backup volume has been filled. Mounting the next volume on the free alternate tape drive ensures an uninterrupted flow of processing.

  When objects are to be restored from backup volumes that precede the currently mounted backup volume, the RESTORE command will request mounting of the volume(s) containing the required data when the volume is needed.

Error Handling

VSE/VSAM Backup/Restore differentiates between object-specific and system-related errors. An object-specific error is shown by a message and the backup or restoration process continues with the next object unless more than 32 object-specific errors have been found.

For system-related errors, the execution of the BACKUP or RESTORE command is terminated immediately.

Backup files whose processing was prematurely terminated can, however, be used for restoration of those objects that were successfully backed up.

RESTORE reports the progress of the total restoration by means of messages. This allows easier back-out when an error occurs.

When the restoration of an object cannot be successfully completed, RESTORE deletes the object again if it was already defined. This ensures that the catalog always reflects the correct status and contains no wrong entries.

Better Performance

Backup/Restore has better performance in execution speed, and has low processor use. This is possible when using backup files on tape, streaming mode, and alternate tape drives. In general, better performance is the result of:

- Selective RESTORE.
- Use of the SVA.
- Variable number and size of buffers.
- Fixing buffers in storage with the UPSI Job Control command.
- DASD cache exploitation.
- Fast KSDS reorganization on CA basis.

Another significant performance factor is the option of *data compaction*. That is, you can write backup files to tape or disk in a compressed format. Data compaction can be requested through the COMPACT parameter in the BACKUP command. (On restoring a compacted backup file, the process automatically corrects the data to normal format.)

Performance Considerations when Changing Files During Restore

Specifying a different use class on the RESTORE command has no appreciable effect on RESTORE performance. When you restore a file to a volume of a different device type or change a file's data allocation information or index CI size, however, it means that the space allocation and blocking characteristics of the restored file will be different from the space allocation and blocking characteristics

of the file that was backed up. In order to process these changes, RESTORE command execution will take longer than when none of these changes is specified. For RRDSs and ESDSs (including SAM ESDS files), streaming mode (if available) of a tape device used as a backup device may be affected. For KSDSs, you can expect a significant performance degradation; streaming mode probably cannot be maintained. The actual degradation depends on device geometry, CA size, and the number of CI splits and record deletions that have occurred.

When to Use EXPORT/IMPORT and when BACKUP/RESTORE

BACKUP/RESTORE operates on the basis of large block processing, thus achieving high performance and low processor use. Therefore, BACKUP/RESTORE should be used for:

- Backing up your files regularly.
- Restoring the backup files when necessary.

EXPORT/IMPORT operates on either a CI or a record basis. Therefore, EXPORT/IMPORT should be used for:

- Migration between systems
- Reorganization at a level lower than CAs. For example, reorganizing on CI basis which includes restoring to a different device type.

The format of a backup file produced by BACKUP differs from the format produced by EXPORT. Therefore, the pairs of commands can not be intermixed. That is, a tape file produced by BACKUP cannot be IMPORTed and a file produced by EXPORT cannot be RESTOREd.

The BACKUP Command (Job Control)

Backup File Characteristics

The BACKUP command creates a backup file on tape or disk volumes. The backup file must always be assigned to SYS005.

Labels Supported by BACKUP

Backup File on Tape

A backup file on tape is created as a labeled or unlabeled file, depending on your specification in the backup job. If you want the backup file to be labeled, you have to provide the TLBL statement in the backup job, and specify the STDLABEL parameter in the BACKUP command (where the STDLABEL specification refers to the TLBL statement).

As tape labels, only standard (EBCDIC) labels are supported. The backup file always starts at the beginning of a volume. When a volume has been filled during backup, the tape is rewound or unloaded.

The backup file is a single- or multivolume tape file consisting of several subfiles separated by tape marks. These tape marks prevent labeled backup files created by BACKUP from being processed as labeled tape files by DTFMT and from sharing a tape volume with other labeled tape files.

On the other hand, there must be a tape mark **at the beginning of every tape volume** (as required by DTFMT) before you can use a brand-new tape volume for backup or change the tape density.

Backup File on Disk Volumes

A backup file on disk volumes can extend over several disk extents on several volumes.

For storing backup files, you can use any supported disk devices, but all disk volumes occupied by a given backup file must be of the same device type.

If the backup file extends over several volumes, the corresponding devices must be assigned to consecutive logical units, beginning with SYS005. The disk extents provided for the backup file must

be defined in the backup job by means of corresponding // EXTENT statements. The sequence in which you specify the statements defines the sequence in which the extents will be used.

Note that a // DLBL statement must precede the EXTENT statements. The DLBL **must** contain the file name **BACKOUT** and a file identification for the backup file. The file type is **SD** (which need not be specified because it is the default).

Job Control Statements for BACKUP

Figure 5 on page 54 shows the job control statements to be used in connection with the BACKUP command.

- It is assumed that the DLBL statement for the master catalog has been placed into the permanent standard label area, as follows:

```
// DLBL IJSYSCT,'VSAM.MASTER.CATALOG',,VSAM
```

- If the file is cataloged in a user catalog, you must provide a DLBL statement identifying that catalog in the format:

```
// DLBL IJSYSUC,'VSAM.USER.CATALOG',,VSAM
```

- The UPSI statement can be used to fix the buffers used by BACKUP in real storage. Where a left-hand bit of:
  - 1 in the UPSI byte indicates that the buffers are to be fixed while being used.
  - 0 (the default) indicates that fixing is not desired.
- *file-id* is the name the backup file will get.

Streaming mode is specified using the mode setting of the IPL ADD command or the ASSGN job control statement.

For information on backing up a file to *disk*, refer to “Backup Example 3” on page 247.

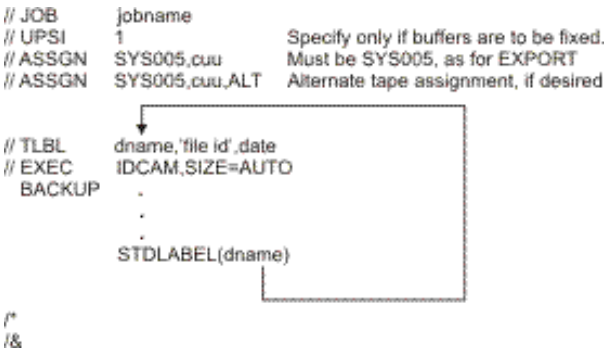


Figure 5. JCL for BACKUP to Tape

The RESTORE Command (Job Control)

Effects

If the object is restored to the same volumes, its new location can (and generally will) be different from the original location. Restoration deletes any old copy of a VSE/VSAM object (if it is defined in the catalog effective for the RESTORE command) regardless of its retention date.

Objects to be restored are automatically defined. That is, you do not need not execute a DEFINE command before restoring the objects. The automatic definition also reflects any modifications you may have specified in the RESTORE command, for example, a new set of volumes to which you wish the objects restored.

RECOVERY and WRITECHECK options, if any, are recorded in the catalog information during backup, but ignored during restoration for the sake of fast file restoration.

Job Control Statements for RESTORE

Figure 6 on page 55 shows the job control statements to be used in connection with the RESTORE command.

- It is assumed that the DLBL statement for the master catalog has been placed into the permanent standard label area with the following job control statement:

```
// DLBL IJSYSCT,'VSAM.MASTER.CATALOG',,VSAM
```

- The UPSI statement can be used to fix the buffers used by Backup/Restore. Where a left-hand bit of:
  - 1 on the UPSI byte indicates that the buffers are to be fixed in real storage while being used.
  - 0 (the default) indicates that fixing is not desired.
- A // DLBL statement must precede the EXTENT statements. Also, the DLBL *must* contain the file name **BACKIN**, and the file type must be **DA**.
- If the backup file resides *on tape*, the tape must be assigned to SYS004.
- If the backup file resides *on disk*, the first volume must be assigned to SYS004. Any subsequent volumes must be assigned to SYS005, SYS006, and so on.
- *file-id* (in the TLBL statement) must be the name of the backup file if it has standard (EBCDIC) tape labels.
- The TLBL statement is only required if the corresponding parameters are specified in the RESTORE command. The volume number should be omitted in the TLBL statement. RESTORE does internal volume sequence verification.

Streaming mode is specified in the mode setting of the IPL ADD command or the ASSGN job control statement.

For information on restoring a file to *disk*, refer to:

- “Labels Supported by BACKUP” on page 53.
- “Restore Example 2” on page 259.

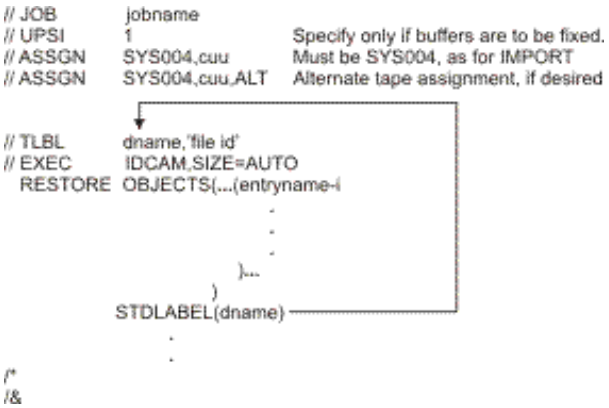


Figure 6. JCL for RESTORE to Tape

IDCAMS Interactive Interface (IDCONS)

Overview

The command component of VSE/VSAM (IDCAMS) cannot be used to input interactive VSAM commands. Unlike other service utilities such as the VSE Librarian program (LIBR), IDCAMS cannot be invoked from the operator's console, accepting input from and routing the output back to the console.

The IDCONS program overcomes this limitation by providing a front-end to allow input of IDCAMS commands from the console. See also "Invoking IDCAMS from a Program" in VSE/VSAM User's Guide and Application Programming.

Because the IDCAMS output is designed for a 120-character-per-line listing output. it may often appear unreadable on an 80 column terminal. The IDCONS program therefore has a option to 'compress' IDCAMS print output by condensing multiple blanks and hyphens into single ones. The drawback, however, is that the column alignment on the output will be destroyed.

Syntax and Usage

IDCONS, which resides in the VSE System Library, can be invoked from the console from ATTN mode or when a PAUSE step is active in the partition. First, you will be prompted as to whether you want to have the IDCAMS output transferred

- back to the VSE console in compressed format
- back to the VSE console in its original format
- to SYSLST rather than the console

You will then be prompted for IDCAMS commands. Normal IDCAMS syntax applies (see “Coding of IDCAMS Commands” on page 4).

IDCONS will supply a leading blank character to satisfy IDCAMS' margin requirements. You can then use up to 72 characters on the line. Handling of continuation lines (the last character is a hyphen) is unchanged. The dialog can be terminated by replying QUIT. Note that this must be entered without leading blanks; hence (for the BG partition, for example) it should look like:

```
0 QUIT
```

Example

The following example illustrates the use of IDCONS (operator input is highlighted):

```
0 exec idcons,size=auto
0 EXEC IDCONS,SIZE=AUTO BG 000 IDCONS400I IDCONS - CONSOLE
INTERFACE TO ACCESS METHOD SERVICES BG 000 IDCONS402I SELECT OUTPUT
DEVICE: BG 000 IDCONS403I 1= SYSLOG (COMPRESSED); 2= SYSLOG
(NORMAL); 3=SYSLST BG 000 IDCONS405D REPLY 1, 2 OR 3 BG-000

0 1
0 1 BG 000 IDCONS406D ENTER IDCAMS COMMAND OR QUIT TO EXIT
BG-000

0 listcat allocation -
0 LISTCAT ALLOCATION - BG 000 IDCONS409D CONTINUE IDCAMS
COMMAND OR QUIT TO EXIT BG-000

0 entries(tst.ksds.cluster) -
0 ENTRIES(TST.KSDS.CLUSTER) - BG 000 IDCONS409D CONTINUE
IDCAMS COMMAND OR QUIT TO EXIT BG-000

0 cat(tst.ucate)
0 CAT(TST.UCATE) BG 000 IDCAMS SYSTEM SERVICES TIME:
08:50:02 06/15/90 PAGE 1 BG 000 BG 000  LISTCAT ALLOCATION
- BG 000  ENTRIES(TST.KSDS.CLUSTER) - BG 000  CAT(TST.UCATE) BG 000
IDCAMS SYSTEM SERVICES TIME: 08:50:02 06/15/90 PAGE 2
BG 000  LISTING FROM CATALOG - TST.UCATE BG 000 CLUSTER -
TST.KSDS.CLUSTER BG 000  HISTORY BG 000  OWNER-IDENT-(NULL)
CREATION-90.166 BG 000  RELEASE-2 EXPIRATION-00.000 BG 000  DATA -
TST.KSDS.DATA BG 000  HISTORY BG 000  OWNER-IDENT-(NULL)
CREATION-90.166 BG 000  RELEASE-2 EXPIRATION-00.000 BG 000  ALLOCATION
BG 000  SPACE-TYPE-TRACK BG 000  SPACE-PRI-8 USECLASS-PRI-0
HI-ALLOC-RBA-147456 BG 000  SPACE-SEC-2 USECLASS-SEC-0
HI-USED-RBA-147456 BG 000  VOLUME BG 000  VOLSER-444444
PHYREC-SIZE-2048 HI-ALLOC-RBA-147456 EXTENT-NUMBER-1 BG 000
DEVTYPE-3380 PHYRECS/TRK-18 HI-USED-RBA-147456
EXTENT-TYPE-X'00' BG 000  VOLFLAG-PRIME TRACKS/CA-2 BG 000
EXTENTS: BG 000  LOW-CCHH-X'00020001' LOW-RBA-0 TRACKS-8
BG 000  HIGH-CCHH-X'00020008' HIGH-RBA-147455 BG 000  INDEX -
TST.KSDS.INDEX ... BG 000 IDC0001I FUNCTION COMPLETED, HIGHEST
```

```
CONDITION CODE WAS 0 BG 000 IDCONS406D ENTER IDCAMS COMMAND OR QUIT TO
EXIT BG-000

0 quit
0 QUIT BG 000 IDCAMS SYSTEM SERVICES TIME:
08:50:02 06/15/90 PAGE 4 BG 000 BG 000 IDC0002I IDCAMS
PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0 BG 000 IDCONS404I
IDCONS COMPLETED.
BG 000 1S55I  LAST RETURN CODE WAS 0000
```

Chapter 2. The IDCAMS Commands

This chapter explains in detail the:

- "Functional Commands and Formats" on page 59
- "Modal Commands and their Formats" on page 201

Functional Commands and Formats

The functional commands discussed in the following are:

ALTER
To alter attributes of files and other objects already defined, "ALTER" on page 60.

BACKUP
To produce a backup copy of VSE/VSAM objects, "BACKUP" on page 72.

BLDINDEX
To build one or more alternate indexes over a base cluster, "BLDINDEX" on page 77.

CANCEL
To cancel the current job or job step, "CANCEL" on page 79.

DEFINE

ALTERNATEINDEX
To define an alternate index, "DEFINE ALTERNATEINDEX" on page 79.

CLUSTER
To define a cluster for a VSE/VSAM file, "DEFINE CLUSTER" on page 97.

MASTERCATALOG
To define the master catalog, "DEFINE MASTERCATALOG" on page 120.

NONVSAM
To define a catalog entry for a nonVSAM file, "DEFINE NONVSAM" on page 129.

PATH
To define a path directly over a base cluster or over an alternate index and its related base cluster, "DEFINE PATH" on page 131.

SPACE
To define a VSE/VSAM data space, "DEFINE SPACE" on page 135.

USERCATALOG
To define a VSE/VSAM user catalog, "DEFINE USERCATALOG" on page 139.

DELETE
To delete files and other objects, including catalogs and nonVSAM files, "DELETE" on page 148.

EXPORT
To create a portable copy of a file and disconnect user catalogs, "EXPORT" on page 155.

IMPORT
To redefine and reload VSE/VSAM files and connect user catalogs, "IMPORT" on page 160.

LISTCAT
To list catalog entries, "LISTCAT" on page 167.

PRINT
To print both VSE/VSAM and nonVSAM files, "PRINT" on page 171.

RECMAP
to map a VSE/VSAM cluster to a relational structure and later maintain the associated map or view"RECMAP" on page 177.

REPRO
To copy and load VSE/VSAM files and catalogs and nonVSAM files, “REPRO” on page 181.

RESTORE
To reinstall the objects of a backup file that was produced through the BACKUP command, “RESTORE” on page 189.

SNAP
To copy Enterprise Storage Server (ESS) volumes, “SNAP” on page 199.

VERIFY
To verify and correct problems that made your file unusable, “VERIFY” on page 200.

In the following, reference is made to *default catalog*. For an explanation, refer to "Search Sequence of Catalogs" in the VSE/VSAM User's Guide and Application Programming.

ALTER

The ALTER command is used to change file attributes in catalog entries. To alter an entry, you need to supply its name and the attributes to be altered. For example, using this command you can change the name and passwords of an object. Any attributes not explicitly specified remain as they were originally defined or last altered.

A key-sequenced file and alternate index consist of three components: an index and data component and the cluster entry itself. An entry-sequenced and relative-record file consist of two components: a data component and the cluster entry. Most attributes of a file are associated with its data or index components. Only retention period, owner ID, and cluster protection attributes are associated with the cluster entry itself; if you specify the cluster name in the entryname parameter, only these attributes can be changed. If you specify the cluster name to alter any attribute not directly associated with the cluster entry, the ALTER request is terminated and an error message is issued. The opposite is also true; if you specify a data or index component name and attempt to alter an attribute directly associated with the cluster entry, your ALTER command is terminated and an error message is issued.

You must specify the explicit data or index component name to alter any of the remaining attributes (such as shareoptions, bufferspace, writecheck, the data or index protection attributes). It is recommended, therefore, that you specify data and index component names at DEFINE time to facilitate the use of the ALTER command. Otherwise, you will have to use the system-generated names. Use the LISTCAT command to determine the system-generated names of the data and index components.

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK.

Catalog Entry Types that Can Be Altered

Table 5 on page 61 lists all the ALTER parameters and the entry types to which every parameter applies. In the figure:

- the **x** indicates that you can respecify the value or attribute for the type of catalog entry. For example, you can specify a new value for the ATTEMPTS parameter for all entry-types except the nonVSAM entry.
- the meaning of the subscripts are as follows:

  [1] You cannot alter a catalog's data or index component entries.
  [2] The data component must be empty.
  [3] The alternate index must be empty.
  [4] It is not permitted to add a new volume to the index component of an indexed or numbered file that was defined with the IMBED attribute.
  [5] Only the name of user catalog (excluding VSESP.USER.CATALOG) can be altered.

Note that if you want to alter information that relates to a user catalog itself (for example, alter the password of a user catalog), you have to either specify the user catalog as the job catalog, or you have to reference the user catalog in the CATALOG parameter of the ALTER command.

Note that altering the name of a user catalog itself requires the specification of USERCATALOG attribute in conjunction with NEWNAME.

For a discussion of ALTER functions and examples, refer to "Altering Catalog Entries" on page 38, and "Example 7: Modifying and Listing the Cataloged Attributes of a File" on page 222.

Table 5. ALTER Parameters and Their Objects

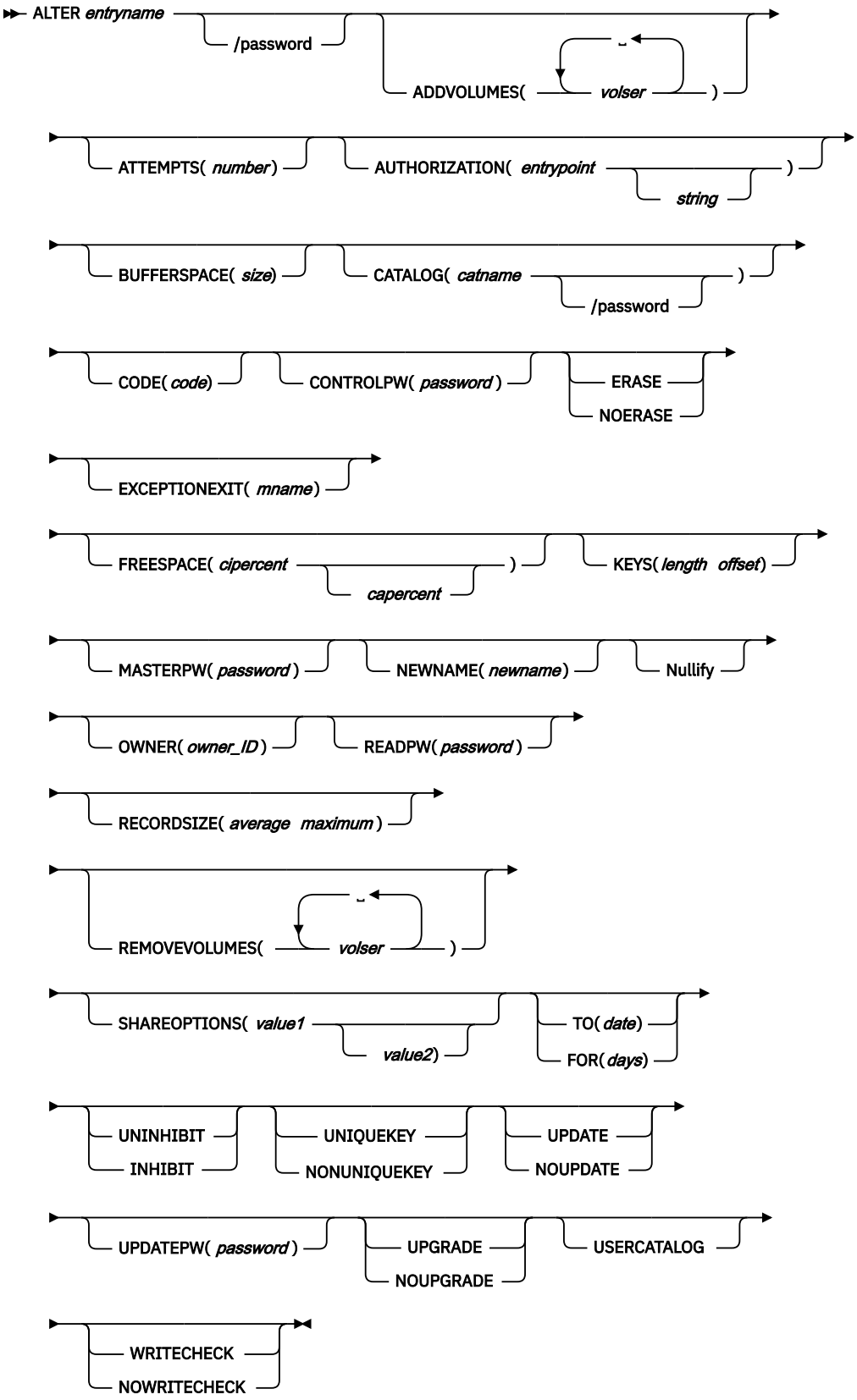
	Type of Catalog Entry								
ALTER Parameters	**Alternate Index**			**Cluster**			**PATH**	**MCAT[1] UCAT[1]**	**NonVSAM**
	AIX	**DATA**	**INDEX**	**CL**	**DATA**	**INDEX**			
ADDVOLUMES.		X	X[4]		X	X[4]			
ATTEMPTS	X	X	X	X	X	X	X	X	
AUTHORIZATION	X	X	X	X	X	X	X	X	
BUFFERSPACE		X			X				
CODE	X	X	X	X	X	X	X	X	
CONTROLPW	X	X	X	X	X	X	X	X	
ERASE		X			X				
EXCEPTIONEXIT		X	X		X	X			
FOR	X			X			X	X	
FREESPACE		X			X				
INHIBIT		X	X		X	X			
KEYS	X[2]	X[2]		X[2]	X[2]				
MASTERPW	X	X	X	X	X	X	X	X	
NEWNAME	X	X	X	X	X	X	X	X[5]	X
NOERASE		X			X				
NONUNIQUEKEY		X							
NOUPDATE							X		
NOUPGRADE	X								
NOWRITECHECK		X	X		X	X			

Table 5. ALTER Parameters and Their Objects (continued)

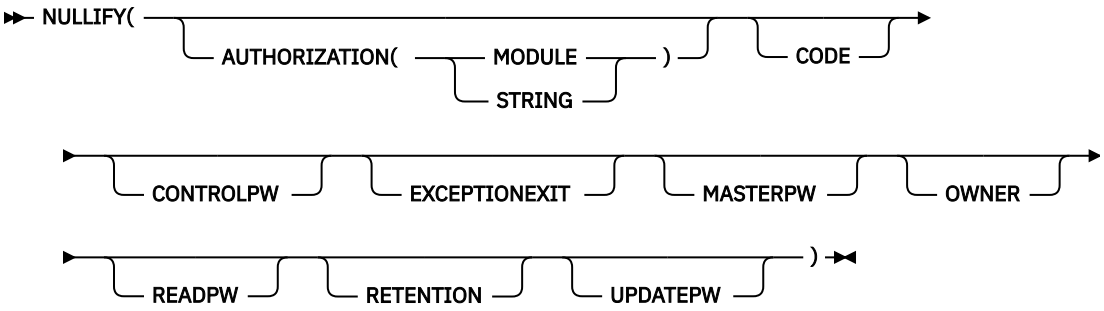
ALTER Parameters	Type of Catalog Entry								
	Alternate Index			Cluster			PATH	MCAT[1] UCAT[1]	NonVSAM
	AIX	DATA	INDEX	CL	DATA	INDEX			
NULLIFY	X	X	X	X	X	X	X	X	
AUTHORIZATION	X	X	X	X	X	X	X	X	
CODE	X	X	X	X	X	X	X	X	
CONTROLPW	X	X	X	X	X	X	X	X	
EXCEPTIONEXIT		X	X		X	X			
MASTERPW	X	X	X	X	X	X	X	X	
MODULE	X	X	X	X	X	X	X	X	
OWNER	X	X	X	X	X	X	X	X	
READPW	X	X	X	X	X	X	X	X	
RETENTION	X			X			X	X	
STRING	X	X	X	X	X	X	X	X	
UPDATEPW	X	X	X	X	X	X	X	X	
OWNER	X	X	X	X	X	X	X	X	
READPW	X	X	X	X	X	X	X	X	
RECORDSIZE	X[2]	X[2]		X[2]	X[2]				
REMOVEVOLUMES		X	X		X	X			
SHAREOPTIONS		X	X		X	X			
TO	X			X			X	X	
UNINHIBIT		X	X		X	X			
UNIQUEKEY		X[2]							
UPDATE							X		

Table 5. ALTER Parameters and Their Objects (continued)									
	Type of Catalog Entry								
ALTER Parameters	**Alternate Index**			**Cluster**			**PATH**	**MCAT[1] UCAT[1]**	**NonVSAM**
	AIX	**DATA**	**INDEX**	**CL**	**DATA**	**INDEX**			
UPDATEPW	X	X	X	X	X	X	X	X	
UPGRADE	X[3]								
WRITECHECK		X	X		X	X			

The format of the ALTER command is:


ALTER entryname
/password
ADDVOLUMES(volser)
ATTEMPTS(number)
AUTHORIZATION(entrypoint string)
BUFFERSPACE(size)
CATALOG(catname /password)
CODE(code)
CONTROLPW(password)
ERASE
NOERASE
EXCEPTIONEXIT(mname)
FREESPACE(cipercent capercent)
KEYS(length offset)
MASTERPW(password)
NEWNAME(newname)
Nullify
OWNER(owner_ID)
READPW(password)
RECORDSIZE(average maximum)
REMOVEVOLUMES(volser)
SHAREOPTIONS(value1 value2)
TO(date)
FOR(days)
UNINHIBIT
INHIBIT
UNIQUEKEY
NONUNIQUEKEY
UPDATE
NOUPDATE
UPDATEPW(password)
UPGRADE
NOUPGRADE
USERCATALOG
WRITECHECK
NOWRITECHECK

Nullify



ALTER Parameters: Summary

Most of the ALTER parameters are the same as those of the DEFINE command. They can be divided into the following categories:

Name, which specifies:

- The entry to be altered (entryname). This parameter is required.
- A new name for the entry (NEWNAME).
- The catalog containing the entry to be altered (CATALOG).
- If a new name should be set for a user catalog, a special attribute (USERCATALOG) should be used together with a NEWNAME parameter.

Data organization, which specifies:

- The length and offset of the key-field (KEYS).
- How a path is to be accessed (UPDATE, NOUPDATE).
- Whether an alternate index is to be kept up to date with its base cluster (UPGRADE, NOUPGRADE).
- Whether an alternate key may belong to several data records in the base cluster (UNIQUEKEY, NONUNIQUEKEY).

Allocation, which specifies:

- The amount of free space to be left in CIs and CAs (FREESPACE)
- The amount of buffer space to be provided (BUFFERSPACE).
- That additional volumes are to be added and removed from the list of candidate volumes (ADDVOLUMES, REMOVEVOLUMES).
- The length of records (RECORDSIZE).

Protection and integrity, which specifies:

- Passwords (MASTERPW, CONTROLPW, UPDATEPW, READPW).
- The protection attributes to be nullified (NULLIFY).
- The type of access available to do certain types of operations (UNINHIBIT, INHIBIT).
- A prompting code (CODE) and number of attempts allowed to provide the correct password in response to prompting at the operator's console (ATTEMPTS).
- A user-supplied authorization verification routine (AUTHORIZATION).
- The owner of the alternate index (OWNER).
- A retention period (FOR, TO) and whether data is to be erased when an entry is deleted (ERASE, NOERASE).
- Whether write-check operations are to be performed as records are inserted (WRITECHECK, NOWRITECHECK).
- The share options to be associated with an entry (SHAREOPTIONS).

- A user-supplied module to be given control when an exception occurs during processing (EXCEPTIONEXIT).

ALTER Parameters in Detail

entryname
is a required parameter that names the object whose entry is to be altered and supplies the master password if the entry defines a password-protected object. Alternatively, the master password of the catalog that contains the entry can be specified in the CATALOG parameter. This parameter must be the first parameter following ALTER.

If KEYS is not specified and if a data or index component entry is to be altered, the master password of the cluster, component, or catalog can be supplied. If KEYS is specified, the master password of the cluster or catalog (but not the component) can be supplied.

If neither KEYS nor RECORDSIZE is specified, the entry that defines the object named is altered.

If KEYS or RECORDSIZE is specified, the entry being altered is as follows:

- If entryname names a path over an alternate index or a base cluster, the entry that defines the data component of the alternate index or base cluster, respectively, whichever is named in PATHENTRY in the path's definition.
- If entryname names an alternate index or a base cluster, the data component of the alternate index or base cluster, respectively.
- If entryname names a data component, the entry of that component.
- If entryname names an index component, an error condition can result. (Entryname can be specified for an index component only if neither KEYS nor RECORDSIZE is specified.)

If any other parameters are specified with KEYS or RECORDSIZE, they will alter the same object as KEYS or RECORDSIZE. For example, if the master password is to be altered (and neither KEYS nor RECORDSIZE is specified) and the entryname names a base cluster, then the master password is altered at the cluster level only. If, however, KEYS or RECORDSIZE/RECORDSIZE is also specified, the master password is altered for the base cluster's data component only.

ADDVOLUMES(volser)
specifies volumes to be added to the list of candidate volumes associated with the data or index component entry being altered. (A candidate volume is reserved for future use by VSE/VSAM.)

A volume number, volser, may contain one through six alphameric, national (@, #, and $), and special characters (commas, blanks, semicolons, parentheses, slashes, asterisks, periods, quotation marks, ampersands, plus signs, hyphens, and equal signs). Single quotation marks within a volume number must be coded as two single quotation marks. A volume number must be enclosed in single quotation marks if it contains a special character. For consistency with VSE job control, only alphameric characters should be used.

Volumes to be added as candidate volumes must already be owned by the catalog that contains the entry being altered; that is, (a) space must have been defined on a volume to be added via the DEFINE SPACE command, or (b) the volume must have been identified as a candidate volume in the DEFINE SPACE command. It is not permitted to add a new volume to the index component of an indexed or numbered file that was defined with the IMBED attribute. You cannot add volumes if the file is currently opened for processing.

Abbreviation: AVOL

ATTEMPTS(number)
changes the maximum number of times (0 through 7) the operator can try to enter a correct password in response to a prompting message. If 0 is specified, the operator is not prompted and is not allowed to enter a password from the console. In this case, if the password has not been supplied through the program, the file will not be opened for processing. This value can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

Abbreviation: ATT

AUTHORIZATION(entrypoint)
specifies (or changes) a user security verification routine (USVR).

entrypoint - specifies the name of the user security verification routine. The name may contain one through eight alphameric, national (@, #, $), or special (hyphen and 12-0 overpunch) characters. The first character must be an alphabetic or a national character. entrypoint is the phase name that appears in the sublibrary.

string - specifies up to 255 characters that are to be passed to the user security verification routine when it receives control to verify authorization. The string must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a string must be coded as two single quotation marks if the string is enclosed in single quotation marks. If the string is specified in hexadecimal, it must be preceded by X and enclosed in single quotation marks (X'string').

Abbreviation: AUTH

BUFFERSPACE(size)
changes the minimum space to be provided for buffers.

The amount specified should be greater than or equal to the amount specified in the original definition. If the amount is less than was specified when the entry was defined, you must ensure that VSE/VSAM has enough space to contain two data component CIs and, if the file is key-sequenced, one index component CI.

Note: Specifications of an invalid value may result in your not being able to restore or import the altered cluster. VSE/VSAM will back up or export the file without error. However, the automatic cluster defined during restore or import will fail due to an invalid buffer size. This may result in an unusable backup tape.

size - is the number of bytes to be provided for buffers. This value can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. If the specified value is less than the amount VSE/VSAM requires, VSE/VSAM gets the amount it requires when the file is opened.

Abbreviations: BUFSP or BUFSPC

CATALOG(catname)
specifies the name of the catalog that contains the entry to be altered. Note that information relating to a user catalog itself is contained in the user catalog (and *not* in the master catalog).

You do not have to specify this parameter (unless it is needed for password specification) when the entry to be altered exists in the default catalog. (The default catalog is (1) the job catalog if one is being used, or (2) the master catalog.)

catname - is the name of the catalog.

password - specifies the master password of the catalog. If the entry to be altered is password-protected and the catalog is also password-protected, you can supply the proper password, either through this parameter or through the entryname parameter. If both passwords are specified, the catalog password is used.

Abbreviation: CAT

CODE(code)
specifies a new code name for the object whose entry is being altered.

The code may contain one through eight EBCDIC characters. The code must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. If a single quotation mark appears within the code, it must be coded as two single quotation marks. Code may also be specified in hexadecimal (X'code') form.

CONTROLPW(password)
specifies a new control password for the object whose entry is being altered.

password - a one through eight EBCDIC character password. If the password contains commas, semicolons, blanks, parentheses, slashes, or asterisks, the password must be enclosed in single

quotation marks. Single quotation marks within a password must be coded as two single quotation marks if the password is enclosed in single quotation marks. Password may also be coded in hexadecimal form (X'password').

Abbreviation: CTLPW

ERASE | NOERASE
specifies (or changes) whether or not the data component whose entry is being altered is to be erased (overwritten with binary zeros) when its entry in the catalog is deleted.

Abbreviations: ERAS and NERAS

Note: ERASE and NOERASE do not function with spanned record files.

EXCEPTIONEXIT(mname)
specifies (or changes) the name of the user module (i.e., the phase name in the sublibrary) to be given control when an exception occurs during the processing of the object whose entry is being altered.

When the exit receives control, it is in the same AMODE that was in effect when the request was issued.

Abbreviation: EEXT

FOR (see TO)

FREESPACE(cipercent)
changes the amount of space that is to be left free after the initial load or any extension and after any split of CIs (cipercent) and CAs (capercent) during sequential processing.

The amounts are specified as percentages which must contain only the numbers 0 to 100 expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. See "Distributed Free Space" on page 21.

Abbreviation: FSPC

INHIBIT (see UNINHIBIT)

KEYS(length offset)
changes the length and offset of the object's key-field.

If the entry being altered is an alternate index, offset applies to the alternate key-field in the data records in the base cluster. For more information, see entryname.

Keys can be specified only if all of the following are true:

- The object whose entry is being altered is an alternate index, a path, a cluster, or a data component of an alternate index or key-sequenced cluster.
- The object whose entry is being altered contains no data records.
- The values for KEYS in the catalog must be default values. However, if nondefault keys are altered and the new value matches the old, processing continues for any other parameters specified on the command.
- The key must fit within the record whose length is specified by the RECORDSIZE parameter.
- The key must fit in the first record segment of a spanned record.
- The new values for KEYS must not conflict with the CI size specified when the object was defined.

length offset - specifies the length of the key-field (between 1 and 255), in bytes, and its displacement from the beginning of the data record, in bytes. You can express length and offset in decimal (n), hexadecimal (X'n'), or binary (B'x') form.

MASTERPW(password)
specifies a new master password for the entry being altered.

See CONTROLPW for the definition of password.

Abbreviation: MRPW

NEWNAME(newname)
specifies a new name for the entry being altered.

The new name may contain 1 through 44 alphameric characters, national characters (@, #, and $), and two special characters (hyphen and 12-0 overpunch). Names that contain more than eight characters must be segmented by periods; one through eight characters may be specified between periods. The first character of any name or name segment must be either an alphabetic or a national character. The last character in the name must not be a period.

If the entry to be altered represents a user catalog, a USERCATALOG attribute must be also specified in order to set a new catalog name.

Abbreviation: NEWNM

NOERASE (see ERASE)

NONUNIQUEKEY (see UNIQUEKEY)

NOUPDATE (see UPDATE)

NOUPGRADE (see UPGRADE)

NOWRITECHECK (see WRITECHECK)

NULLIFY

The protection attributes identified by the subparameters of NULLIFY are to be nullified. Any attribute specified in NULLIFY is nullified before attributes specified in other parameters take effect.

Abbreviation: NULL

AUTHORIZATION(MODULE | STRING)

Either the user authorization routine (MODULE) or the user authorization record (STRING) is to be nullified.

When MODULE is specified, the module name is removed from the catalog record, but the module itself is not deleted. If you nullify the user authorization module, the user authorization record (character string) is also nullified. If, however, you nullify the authorization record, the corresponding module is not nullified.

Abbreviations: AUTH, MDLE, and STRG

CODE

The code name used for prompting is to be nullified.

CONTROLPW

The control password is to be nullified.

Abbreviation: CTLPW

EXCEPTIONEXIT

The exception exit is to be nullified.

Abbreviation: EEXT

MASTERPW

The master password is to be nullified. If a new master password is not specified and if other passwords exist, the highest-level existing password automatically becomes the master password.

Abbreviation: MRPW

OWNER

The owner identification is to be nullified.

READPW

The read password is to be nullified.

Abbreviation: RDPW

RETENTION

The retention period, specified in a TO or FOR parameter (of a DEFINE or ALTER command), is to be nullified.

Abbreviation: RETN

UPDATEPW
The update password is to be nullified.

Abbreviation: UPDPW

OWNER(owner ID)
The new owner identification of the entry being altered.

The owner ID may contain one through eight EBCDIC characters. The owner ID must be enclosed in single quotation marks if it contains a comma, semicolon, blank, parenthesis, slash or slash/asterisk combination. If a single quotation mark appears within owner ID, it must be coded as two single quotation marks when the owner ID is enclosed in single quotation marks. Owner ID can also be expressed in hexadecimal form (X'owner ID').

READPW(password)
specifies a new read password for the object whose entry is being altered.

See CONTROLPW for the definition of password.

Abbreviation: RDPW

RECORDSIZE(average maximum)
specifies the new average and maximum lengths for the data records in the object.

RECORDSIZE can be specified only if all of the following are true:

- The object whose entry is being altered is an alternate index, a cluster, a path, or a data component of a key-sequenced cluster or alternate index.
- The object whose entry is being altered contains no data records.
- The value for maximum RECORDSIZE in the catalog is the default value. However, if a nondefault value is altered and the new value matches the old, processing continues for any other parameters specified on the command. See the DEFINE command (that defines the object) for the default value.
- The new maximum record length is at least seven bytes less the than CI size, unless the record is a spanned record.
- For spanned records, the maximum record size does not exceed CA size minus 10 bytes of control information for every CI in the CA. The average record size does not exceed 32,758 bytes (maximum CI size minus 10 bytes of control information).
- The new record length is large enough to contain all prime and alternate keys previously defined.
- For an alternate index, if NONUNIQUEKEY is specified, RECORDSIZE/RECORDSIZE accounts for the increased record size resulting from the multiple prime key pointers in the alternate index data record.

Note: (See Table 5 on page 61 for applicable entry-types to be altered.) If the object whose entry is being altered is an alternate index path, the alternate index itself is altered; if the path points directly to its base cluster, then the base cluster is altered. If the object whose entry is being altered is an alternate index, the alternate key must be within the limit specified by maximum. For more information, see entryname.

Abbreviation: RECSZ

REMOVEVOLUMES(volser)
specifies volumes to be removed from the list of candidate volumes associated with the data or index component entry being altered.

Volumes specified are removed after any new volumes are added to the volume list (by way of ADDVOLUMES). If a volume to be removed contains data that belongs to the entry being altered, the volume is not removed and an error message is issued. You cannot remove the last (only) volume of a NOALLOCATION file component even if the volume is indicated as a candidate. See ADDVOLUMES for information on how to code volser. Abbreviation: RVOL

SHAREOPTIONS(value1 value2)
specifies a new share option value for a file (data or index component of a cluster or alternate index). Note that you cannot alter SHAREOPTIONS values of a file if the file is currently open for processing by another program.

value1 specifies the degree of file sharing that is to be allowed for input and output processing. Every opened ACB counts as one 'request'.

value2 is used only in conjunction with *value1 = 4*, and specifies the degree of file sharing that is to be allowed for output processing from multiple z/VSE systems. Every opened ACB counts as one 'request'. In *value2*, you can specify 1, 2, 3, or 4. The values 1, 2, and 3 are for purposes of OS/VS compatibility only.

For a more detailed description of the various share options, see the "DEFINE CLUSTER Parameters" on page 102 or "DEFINE ALTERNATEINDEX Parameters" on page 83.

Abbreviation: SHR

TO(date) | FOR(days)
specifies the new retention period for the entry being altered.

TO(date)
specifies the new date until which the entry is to be kept. The date has the form yyyyddd, where yyyy (GE current date through 2099) is the year and ddd (001 through 366) is the day. The new expiration date cannot be a past date, and cannot be more than 99 years to the future. The old date form yyddd is also allowed for compatibility reasons. The yy value will be implicitly converted into a yyyy value, so that the date relates to the future. A value of the *yyddd* portion greater than 99365 (such as 99999, 1999999, or 2099999) will be considered to mean "retain indefinitely, does not expire".

FOR(days)
specifies the number of days for which the entry is to be kept. If the number specified is 0 through 1830, the cluster is retained for the number of days specified; if the number is 1831 through 9999 (maximum), the expiration date of the cluster is considered as a never-expire date. This value can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

UNINHIBIT | INHIBIT
specifies whether the entry being altered can be accessed for any operation or for read operations only.

UNINHIBIT specifies that the read-only restriction set by a previous ALTER or EXPORT/EXPORT command is to be removed. INHIBIT says the object defined by the entry being altered is only to be read.

Abbreviations: UNINH and INH

UNIQUEKEY | NONUNIQUEKEY
specifies whether the alternate-key value can be found in more than one of the base cluster's data records.

UNIQUEKEY indicates that every alternate-key value is in one and only one data record in the base cluster. NONUNIQUEKEY indicates that an alternate key may be in several records in the base cluster.

UNIQUEKEY can be specified only for an empty data component of an alternate index (a component that contains no data records).

The data component need not be empty to specify NONUNIQUEKEY. When NONUNIQUEKEY is specified, respecify the RECORDSIZE parameter to account for the increased record size resulting from multiple prime key pointers in the alternate index data record. Note that you may be required to rebuild the alternate index because RECORDSIZE can only be specified for an empty file.

Abbreviations: UNQK and NUNQK

UPDATE | NOUPDATE
specifies whether or not the upgrade set belonging to the base cluster of the path is to be updated.

Abbreviations: UPD and NUPD

UPDATEPW(password)
specifies a new update password for the object whose entry is altered.

See CONTROLPW for the definition of password.

Abbreviation: UPDPW

UPGRADE | NOUPGRADE
specifies whether VSE/VSAM is to keep the alternate index up-to-date with its base cluster or you assume responsibility to do so.

UPGRADE indicates that whenever a change to the base cluster (other than through CI access) calls for updating the alternate index, VSE/VSAM is to do it. IF NOUPGRADE is specified, you are responsible for maintaining the alternate index.

If the base cluster is open when the ALTER command is issued, the UPGRADE or NOUPGRADE attribute is not effective until the next open of the base cluster, which creates a new set of control blocks.

UPGRADE can be specified only for an empty alternate index. (one that is defined but not built). NOUPGRADE can be specified for an alternate index at any time.

Abbreviations: UPG and NUPG

USERCATALOG
specifies that the object to be renamed is a user catalog. This parameter must be specified if you want to alter a name of a user catalog (to be used together with a NEWNAME parameter). A user catalog can only be renamed if it is not in use by any other partition (or system). Changing the name of the master catalog (IJSYSCT) or the default system user catalog (VSESPUC) is not allowed.

Note: If the user catalog name is changed, all existing labels referring the old catalog name will point to non-existing catalog and thus fail when referred to by the application. The user should take care of such labels separately (ALTER does not perform this processing automatically).

Abbreviations: UCAT

WRITECHECK | NOWRITECHECK
specifies whether to check the data transfer of records written in the object.

Abbreviations: WCK and NWCK

BACKUP

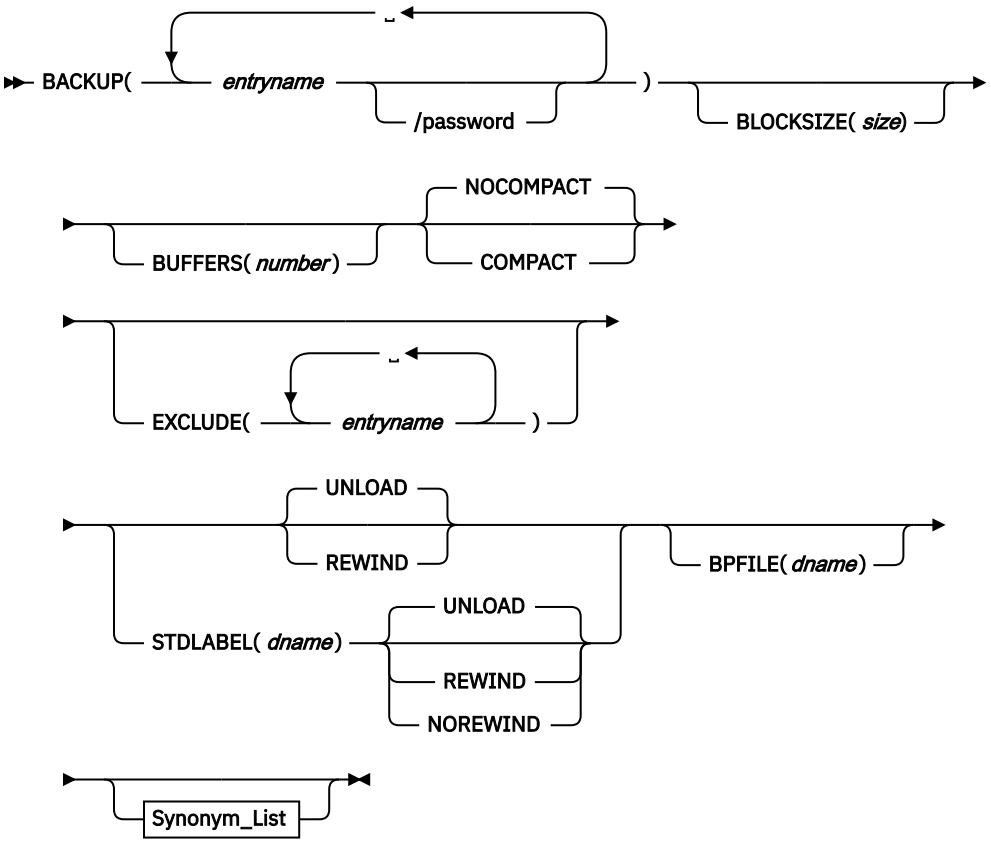
The BACKUP command (abbreviated form: BUP) is used to produce a backup copy of one or more of the following VSE/VSAM objects:

- Key-sequenced data set (KSDS)
- Entry-sequenced data set (ESDS)
- Relative-record data set (RRDS)
- Variable-length relative-record data set (VRDS)
- Alternate index (AIX)
- SAM ESDS file in CI format
- Paths
- Enterprise Storage Server (ESS) volumes

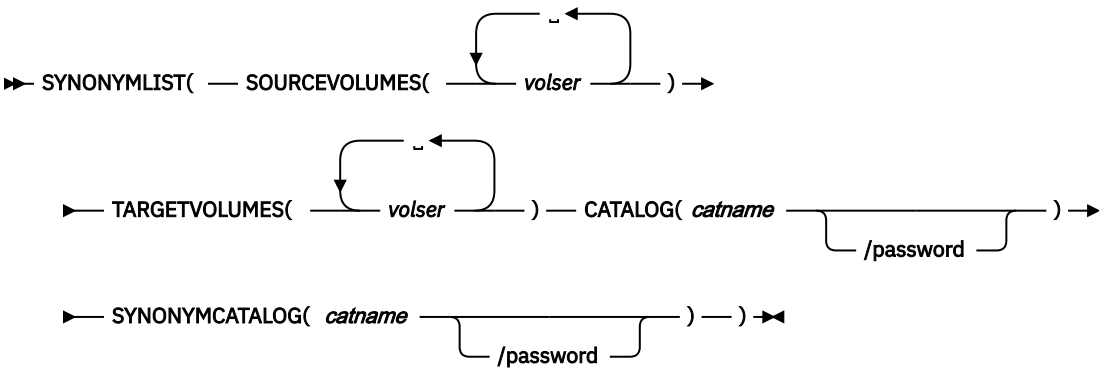
The object to be backed up may be empty. (An empty object is one that was defined using the NOALLOCATION parameter, has never been loaded with data, or has not been loaded since being reset.) All file options that can be specified in the IDCAMS command DEFINE are supported for VSE/VSAM objects. The RECOVERY and WRITECHECK options are recorded in the catalog information, but ignored during restoration for the sake of fast file restoration.

When an object is backed up, its associations (alternate indexes and paths) are automatically backed up with it unless explicitly excluded.

The format of the BACKUP command is:



Synonym_List



For more information and examples, see “Using BACKUP and RESTORE” on page 48 and “Examples of Functions for BACKUP” on page 240.

Abbreviation: BUP

BACKUP Parameters

(entryname)
names the object(s) to be backed up.

- entryname - is the name of an object or set of objects that are candidates for backup. It can be either the full object name or a generic name.

A generic name is an entryname that contains an asterisk (*) as the last name segment. It represents all entrynames (including empty objects) contained in the default (job or master) catalog that start with the name segment(s) preceding the * in the generic name. Specify one or more complete name segments; be sure to include the final period before the asterisk.

- Note that a generic name is an abbreviation for a set of explicitly named objects.
- If only one entryname is specified, the parentheses surrounding the entryname/password may be omitted.
- If you want to back up all objects of the catalog, specify * for entryname.
- You can specify up to 255 entrynames with a single command. This number is not reduced by any alternate indexes and paths, which are automatically backed up with their related base clusters or alternate indexes.
- password - specifies the password of the object to be backed up if it is password-protected. The password specified must be the file or catalog master password if the protection information of the object is to be backed up with the object for reinstallment during restoration. If the associated entryname is a generic name, all objects represented by the generic name must have the same master password unless the master password of the default job or master catalog is specified.
- If the password specified is not the master password, the object(s) will be backed up without password and a message is issued to indicate this fact. The minimum password you must specify is the file's read password. Otherwise the object(s) will not be backed up. If the read password for an object indicated by a generic name is specified, all objects represented by the generic name must have the same read password.

BLOCKSIZE(size)

specifies the size (in bytes) of the buffers to be written to the backup file. The buffer size must be between 512 bytes and 65,535 bytes (64KB minus 1). A buffer of specified size must accommodate an integral number of physical blocks of the object's data component. For key-sequenced files, the buffer size must be at least as large as the index CI size.

Considerations for a Backup File on Magnetic Tape

The specified BLOCKSIZE value is an approximate value for buffer size. For every object to be backed up, this value is rounded, if necessary, to meet the conditions mentioned above. If the backup device supports streaming mode, the minimum buffer size required for streaming is enforced.

If you do not specify a BLOCKSIZE value, VSE/VSAM Backup/Restore determines the buffer size based on the device characteristics, the physical characteristics of the file, and the minimum buffer size requirements for streaming mode (if available and used).

Usually the blocksize for tape devices should be chosen as close as possible to its maximum of 65535 (64K-1). This minimizes the number of I/O operations that need to be done. However, some older tape devices (such as 3420 or 0621) should be operated with a blocksize of 32768.

Considerations for Tape Devices in Streaming Mode

Because the buffer size determines the size of the tape blocks, it also determines the number of tape I/O operations that are needed for backing up or restoring an object. Increasing the buffer size also increases the time available to VSE/VSAM Backup/Restore for re-instructions to the tape device, which enables you to positively influence the system's streaming capability under different workload conditions. The larger the specified BLOCKSIZE value, the more likely that streaming is achieved and the less processor time is required per transferred byte, thus increasing the multiprogramming capability.

Considerations for a Backup File on Disk Volumes

For every object to be backed up, it is checked if the buffer size value specified in BLOCKSIZE complies with the conditions mentioned above. If it complies, the buffer size value is used. If it does not comply, or if the BLOCKSIZE value is not specified, VSE/VSAM Backup/Restore determines the buffer size based on the device characteristics of the backup file and the physical characteristics of the VSE/VSAM file. For CKD and ECKD devices, a buffer size value of about half of the track capacity is favored; for FBA devices, the buffer size value is at least as large as four times the size of the index CI.

Abbreviation: BLKSZ

BPFILE(dname)
indicates that the backup file resides on disk.

dname specifies the *filename* of the DLBL statement that contains the necessary label and extent information. If the BPFILE parameter is not specified, and SYS005 is assigned to a disk device, a dname of BACKOUT is assumed.

BUFFERS(number)
specifies the number of common data buffers to be used for the backup operation. (The size of every buffer is derived from the BLOCKSIZE specification.) By means of this parameter, you can vary the performance behavior of the backup operation in a multiprogramming environment and influence the streaming ability of the backup function. Increasing the number of buffers increases the streaming potential.

The minimum number of buffers that can be specified is 2, the default is 3, and the maximum is 255. If data compaction (see COMPACT parameter) is requested, the minimum number of buffers that can be specified is 3. Specifying a very large number of buffers does not further improve performance. Usually a value up to 8 makes sense.

Note that backup will write the number of buffers specified in the BUFFERS parameter following the trailing reflective marker of a tape volume. Therefore, the marker must be far enough from the end of the tape volume to allow space for the buffers. If block size and number of buffers are too large, backup fails and, depending on the type of tape device, message 0P10E EQUIP CHK or message 0P65I MEDIA ERR is issued.

The // UPSI statement can be used to fix the buffers. For more information, refer to “Job Control Statements for BACKUP” on page 54.

Abbreviation: BFRS

COMPACT | NOCOMPACT
specifies data compaction for backup files. Specifying COMPACT means that all data in a backup file is to be compacted before being written to the backup volumes. Data compaction may reduce the:

- Space required on the backup volume.
- Number of I/O operations for writing the backup file.

Thus, performance of the backup process can be increased. On restoring a compacted backup file, the process automatically changes the data to normal format.

Abbreviations: CO and NCO

Default: NOCOMPACT

EXCLUDE(entryname)
specifies a list of VSE/VSAM objects that are not to be backed up, though they may be named (for example, via a generic name) in the list of VSE/VSAM objects for backup.

entryname - may be the name of a single object or a generic name. Up to 255 entrynames can be specified. You may specify the names of empty objects to be excluded.

If an object is excluded from backup, any alternate indexes and paths defined for the object are also excluded, unless they (the alternate indexes) are explicitly named in the list of backup objects.

Note that a generic name is an abbreviation for a set of explicitly named objects. That is to say, all objects comprised by the generic name are interpreted as being explicitly named. In particular, an entry name of "*" explicitly names an alternate index.

If an object is backed up, but an alternate index defined for it is excluded from backup, the alternate index (and any paths defined for it) are automatically deleted when the object is restored.

Abbreviation: EXCL

STDLABEL(dname)
is required if the backup file is to be created on tape volumes with (EBCDIC) tape labels, to prevent accidental destruction of other labeled tape files or of the backup file itself. It is also required for multiple catalog backups (see NOREWIND).

dname - specifies the filename of the TLBL statement that contains the necessary label information.

Abbreviation: SLBL

REWIND | UNLOAD
REWIND specifies that the backup file tapes are to be rewound to load point but not unloaded on an OPEN, CLOSE, or EOV condition.

Abbreviation: REW

UNLOAD specifies that the backup file tapes are rewound on an OPEN, and rewound and unloaded on an EOV and CLOSE condition. See also note for NOREWIND.

Abbreviation: UNLD

Default: UNLOAD

See also NOREWIND.

NOREWIND
BACKUP keeps the tape positioned after the step has finished. This is then the start position for the next BACKUP (possibly of another catalog) to the same tape in a subsequent job step. This option is supported only in conjunction with STDLABEL.

Note: If a backup job is for more than one catalog, the NOREWIND parameter must be used for *all* job steps, including the last. You cannot use REWIND or UNLOAD within such a job stream.

Abbreviation: NREW

SYNONYMLIST
Indicates that this backup uses a synonym list of "snapped" VSAM volumes (see “Backup Example 8” on page 252).

Abbreviation: SYNLIST or SYNL

SOURCEVOLUMES(volser[volser...])
List indicating the volumes of which a snapshot is to be taken.

Abbreviation: SVOLUME or SVOL

TARGETVOLUMES(volser[volser...])
List indicating to which a snapshot is to be written.

Abbreviation: TVOLUME or TVOL

CATALOG(catname[/password])
Specifies the name and password of the source catalog, which is the original catalog from which a snapshot was done.

Abbreviation: CAT

SYNONYMCATALOG(catname[/password])
Specifies the synonym name and password of the snapped (target) catalog, which was copied (using IXFP and FlashCopy) to the snapped (target) volume. You must ensure that the synonym name of the catalog has been imported using IMPORT CONNECT before running the IDCAMS BACKUP. The passwort is identical to the password of the source catalog.

Note: You must provide the synonym name of the catalog in a DLBL statement containing filename IJSYSUC.

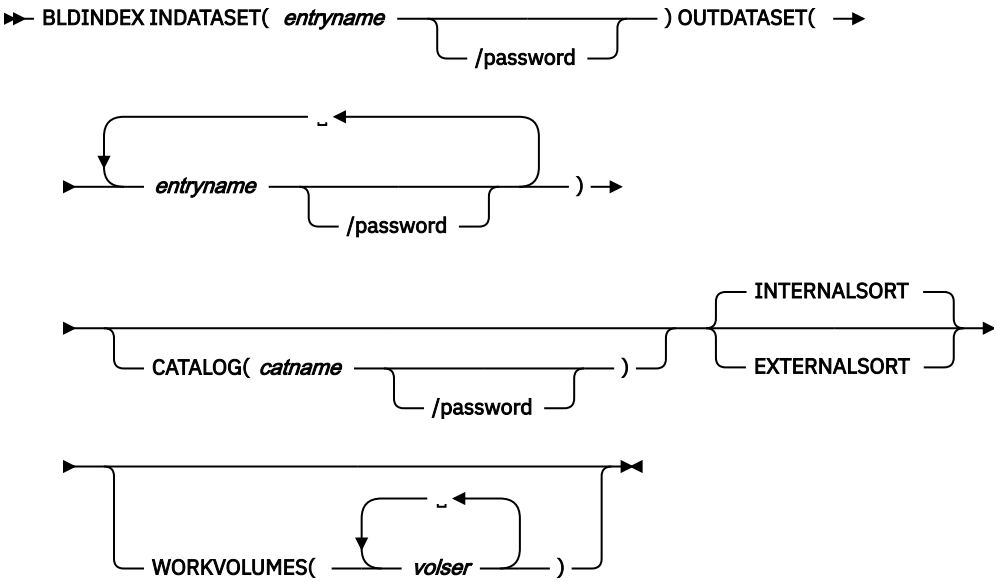
Abbreviation: SYNCAT

BLDINDEX

The BLDINDEX command (BIX) is used to build one or more alternate indexes over a base cluster. Both the alternate index and the base cluster must have been previously defined as related to each other by a path, and the base cluster must have been loaded. The base cluster over which the alternate index is built must contain either an entry- or a key-sequenced file, and this file cannot be reusable. See "Building an Alternate Index" on page 34 and "Example 8: Creating an Alternate Index and Its Path" on page 223 for a discussion of BLDINDEX functions and examples.

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under "PARM" on page 205.)

The format of the BLDINDEX command is:



BLDINDEX Parameters

CATALOG(catname)

specifies the name and the update password, if required, of the catalog in which sort work files are to be defined if an external sort is performed. (For further information on sorting, see "Building an Alternate Index" on page 34.) Table 6 on page 77 shows when to specify the CATALOG parameter and a DLBL statement for a job catalog (IJSYSUC) and, when applicable, a user catalog (not IJSYSUC). A DLBL statement (specifying IJSYSCT) for the master catalog is always required.

Table 6. When to Specify DLBL and CATALOG Parameter

Location of Alternate index ==> Work file ==>	**MCAT MCAT**	**UCAT MCAT**	**MCAT UCAT1**	**UCAT1 UCAT2**	**MCAT none**	**UCAT none**
Specify job cat DLBL?	no	yes	no	yes (UCAT1)	no	yes
Specify BLDINDEX CATALOG parameter?	no**	yes (MCAT)*	yes (UCAT1)*	yes (UCAT2)*	no**	no**
*) If you specify the WORKFILES parameter, you must specify CAT= on the DLBL statement. Specify the WORKVOLUMES parameter instead, because it does not require a DLBL for the work file.						

Table 6. When to Specify DLBL and CATALOG Parameter (continued)						
Location of Alternate index ==> Work file ==>	**MCAT MCAT**	**UCAT MCAT**	**MCAT UCAT1**	**UCAT1 UCAT2**	**MCAT none**	**UCAT none**
**) Unless a password is required, in which case you must specify the CATALOG parameter.						

Abbreviation: CAT

EXTERNALSORT | **INTERNALSORT**
EXTERNALSORT indicates that BLDINDEX is to perform an external sort *if insufficient GETVIS space is available for an internal sort*. For an external sort, see the WORKVOLUMES parameter. BLDINDEX dynamically defines two VSE/VSAM files and uses them as work files for the external sort. The sort work files are deleted at the end of the sort.

When EXTERNALSORT is specified, but sufficient GETVIS space is available for internal sorting, BLDINDEX performs an internal sort.

When INTERNALSORT is specified, BLDINDEX performs an internal sort *if sufficient GETVIS space is available*. (In that case, the WORKVOLUMES parameter is not required.) If not enough GETVIS space is available, BLDINDEX forces an external sort. (For further information on sorting, see "Building an Alternate Index" on page 34.)

Abbreviations: ESORT and ISORT

Default: INTERNALSORT

INDATASET(entryname)
specifies the entryname of (a) the base cluster or (b) the path that points to the base cluster. The entry must be in the default catalog. If the base cluster is password-protected, you may give any one of its passwords. If a path over the base cluster is specified, the password must be that of the path. This parameter is required.

Abbreviation: IDS

OUTDATASET(entryname)
specifies the entryname of the alternate index or the path that points to the alternate index. If the alternate index is password-protected, you must give its update or higher-level password. If a path over the alternate index is specified, the password must be that of the path. More than one alternate index may be built simultaneously by specifying multiple entryname subparameters. The entries must be in the default catalog. This parameter is required.

Abbreviation: ODS

WORKVOLUMES(volser)
specifies up to ten volume numbers on which to suballocate space for two work files if an external sort is to be performed. However, if enough GETVIS space is available for internal sorting, BLDINDEX performs an internal sort, and volumes for the work files are not required. If not enough space is available, BLDINDEX performs an external sort.

The work file will be created in class 0 space in the catalog specified in the CATALOG parameter or, if that parameter is omitted, in the default catalog.

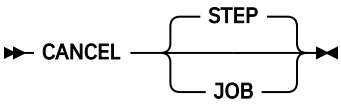
You can omit this parameter if an entry-sequenced file default model exists in the catalog. In this case, the needed volume(s) are selected from the volume list of the default model. The sort work files are deleted at the end of the sort.

Abbreviation: WVOL

CANCEL

The CANCEL command allows you to cancel either the current job or job step. You can use this command along with the modal commands, IF-THEN-ELSE, to conditionally terminate the current job or job step. The CANCEL command will issue an appropriate termination message. A CANCEL STEP will return with code 0. The MAXCC will be the highest return code of any preceding commands.

The format of the CANCEL command is:



CANCEL Parameters

STEP|JOB

STEP
cancels the current job step.

JOB
cancels the current job.

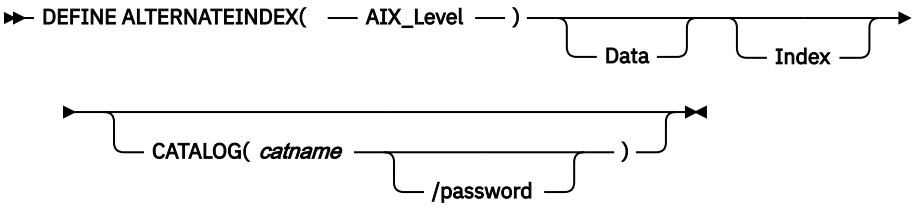
Default: STEP

DEFINE ALTERNATEINDEX

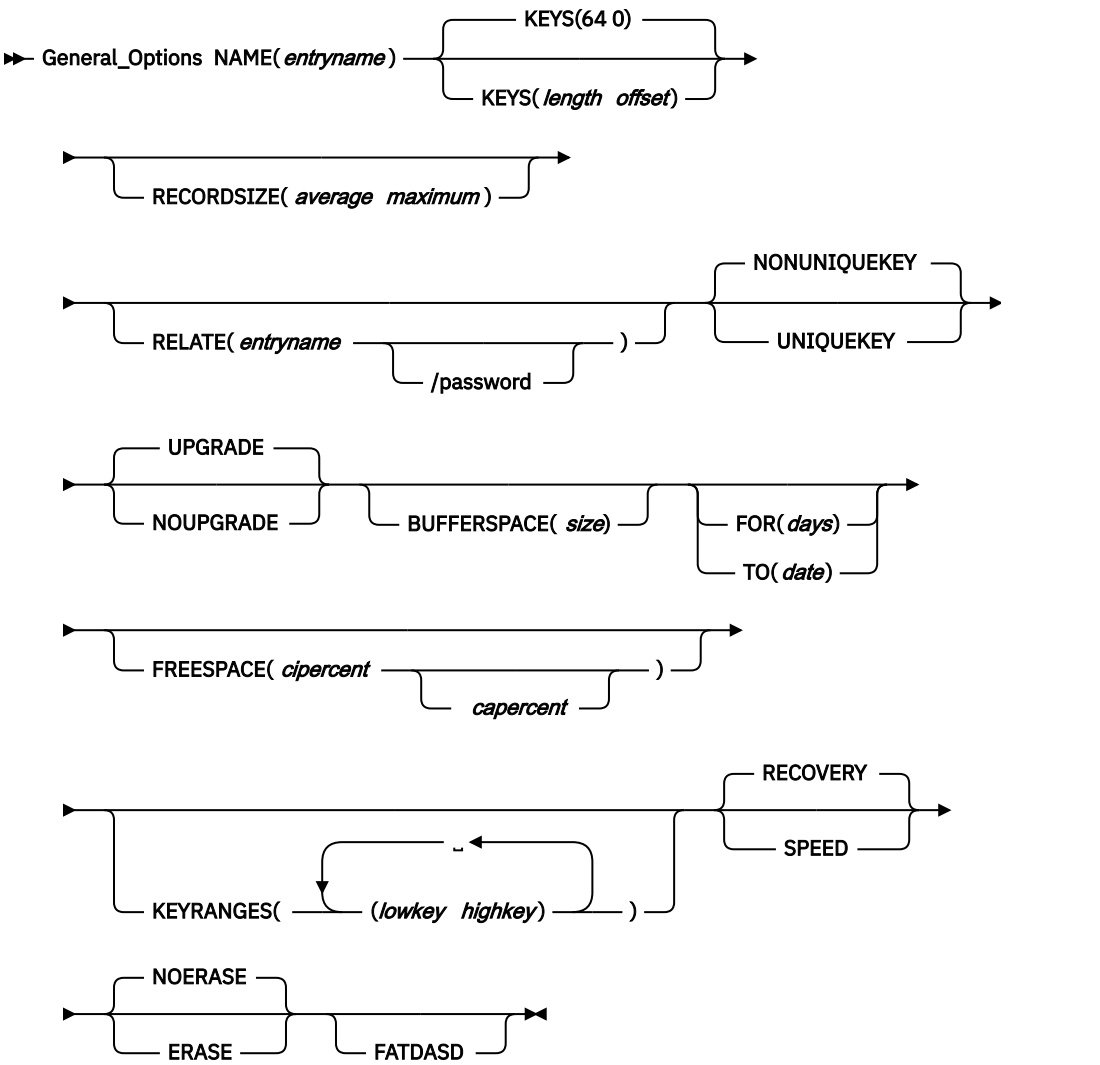
The DEFINE ALTERNATEINDEX command (DEF AIX) defines and relates an alternate index to an existing VSE/VSAM file (base cluster). The base cluster over which the alternate index is built may contain either an entry- or a key-sequenced file. An alternate index cannot be built over a relative-record or VRDS file, another alternate index, a catalog, or a reusable or empty cluster. The alternate index is built using the BLDINDEX command and kept up to date, if desired, by VSE/VSAM. An alternate index is a spanned, key-sequenced file. See “Defining an Alternate Index” on page 33 and “Example 8: Creating an Alternate Index and Its Path” on page 223 for examples and more information.

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (for further information, see the description of the SYNCHK parameter under “PARM” on page 205.)

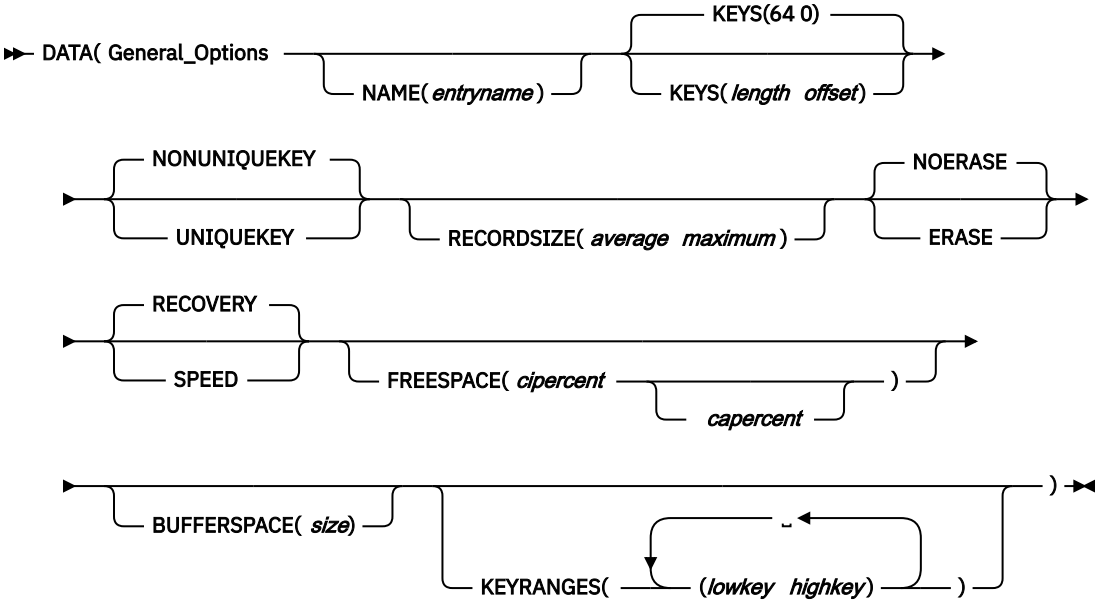
The format of the DEFINE ALTERNATEINDEX command is shown below. Note that the organization of the command consists of three groupings. These groupings are referred to as levels in this publication. Note that some parameters appear in several levels. (The optional CATALOG parameter stands alone; it does not belong to any level.)



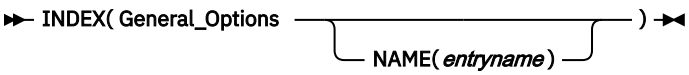
AIX_Level


General_Options
NAME(entryname)
KEYS(64 0)
KEYS(length offset)
RECORDSIZE(average maximum)
RELATE(entryname
/password
)
NONUNIQUEKEY
UNIQUEKEY
UPGRADE
NOUPGRADE
BUFFERSPACE(size)
FOR(days)
TO(date)
FREESPACE(cipercent
capercent
)
RECOVERY
SPEED
KEYRANGES(
(lowkey highkey)
NOERASE
ERASE
FATDASD

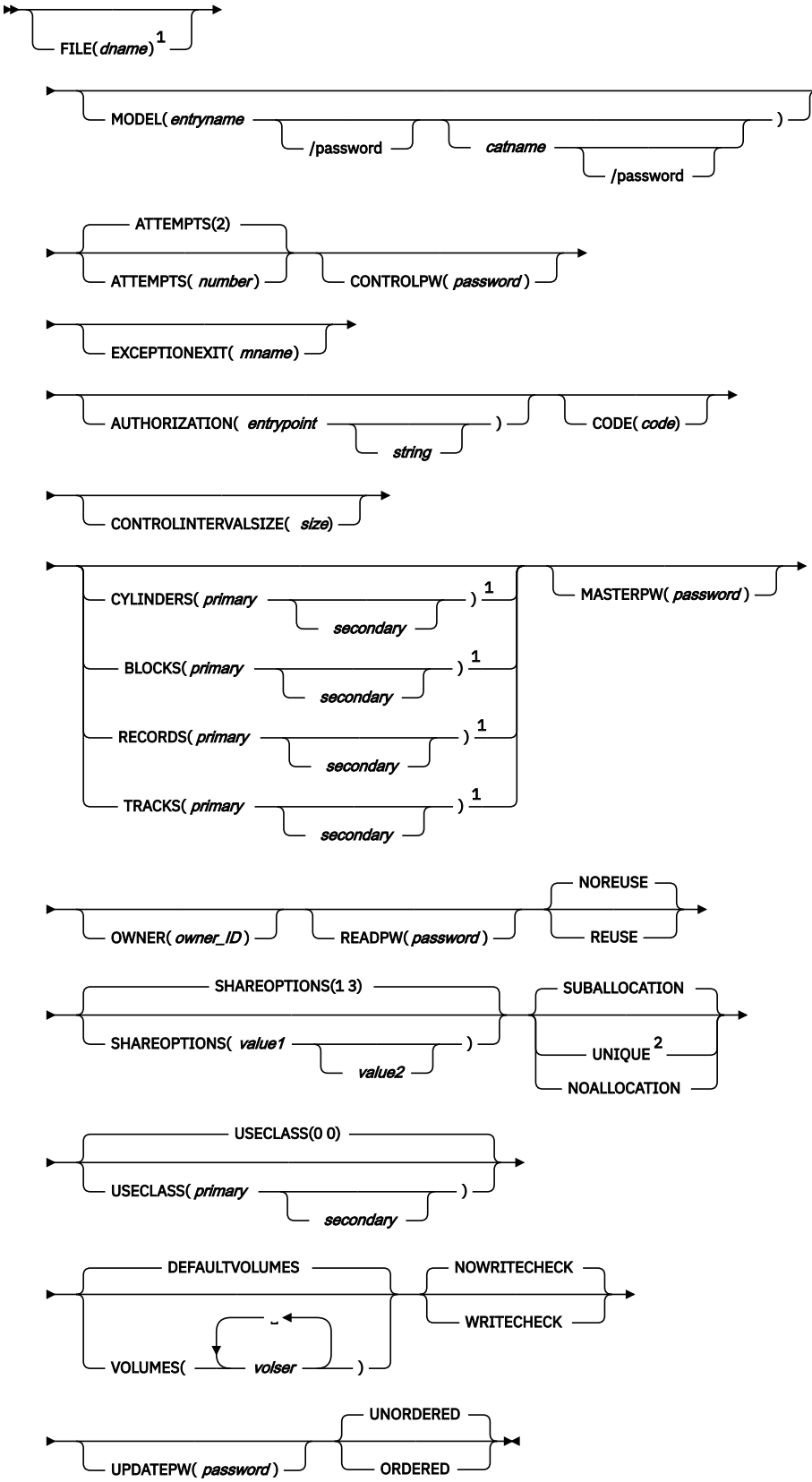
Data


DATA(General_Options
NAME(entryname)
KEYS(64 0)
KEYS(length offset)
NONUNIQUEKEY
UNIQUEKEY
RECORDSIZE(average maximum)
NOERASE
ERASE
RECOVERY
SPEED
FREESPACE(cipercent
capercent
)
BUFFERSPACE(size)
KEYRANGES(
(lowkey highkey)
)

Index



General_Options



Notes:

[1] This parameter is required under certain conditions.
[2] This parameter is invalid under certain conditions.

DEFINE ALTERNATEINDEX Parameters: Summary

The parameters of the DEF AIX command can be divided into the following categories:

Name, which specifies:

- The name of the alternate index (NAME) and, optionally, of its components. This parameter is required.
- The name of the catalog in which the alternate index is to be defined (CATALOG).
- The name of the alternate index to be used as a model for this alternate index (MODEL). For more information, refer to "Using an Object as a Model" in the VSE/VSAM User's Guide and Application Programming.
- The identity of the base cluster over which the alternate index is to be built (RELATE). This parameter is required except for default models.

Data organization, which specifies:

- Length and offset of the alternate-key field in the base cluster (KEYS).
- Whether the alternate index is to be reusable (REUSE, NOREUSE).
- Whether the alternate-key value may belong to several records in the base cluster (UNIQUEKEY, NONUNIQUEKEY).
- Whether VSE/VSAM is to upgrade the alternate index every time its base cluster is modified (UPGRADE, NOUPGRADE).

Allocation, which specifies:

- The volume(s) used for space allocation (DEFAULTVOLUMES, VOLUMES). FILE must be specified to provide data space extents only if UNIQUE is specified. VOLUMES or DEFAULTVOLUMES can be specified at the alternate index level, data component level, or data and index component level.
- The amount of space to be allocated (BLOCKS, CYLINDERS, RECORDS/RECORDS, TRACKS). One of these parameters is required (unless MODEL is specified).
- Whether an alternate index is to reside in either a unique or previously-defined data space (UNIQUE, SUBALLOCATION), or have no space allocated to it (NOALLOCATION).
- The particular class of space to be suballocated to the alternate index (USECLASS). For more information, refer to "Data Space Classification" in the VSE/VSAM User's Guide and Application Programming.
- The percentage of free space to be distributed throughout the data component of an alternate index (FREESPACE).
- Whether the data is to be ranged by key across volumes (KEYRANGES) and, if so, whether the volumes are to be allocated in the order specified (ORDERED, UNORDERED).
- The space to be provided for buffers (BUFFERSPACE) and the size of CIs (CONTROLINTERVALSIZE).
- The average and maximum lengths of data records in the alternate index (RECORDSIZE).
- The FATDASD parameter, it has to be explicitly specified, otherwise VSAM handles a DASD with more than 64K tracks as BIG-DASD (limited to 10017 cylinders).

Protection and integrity, which specifies:

- The passwords to be associated with the alternate index or its components (MASTERPW, CONTROLPW, UPDATEPW, READPW).
- A prompting code (CODE) and the number of attempts allowed to provide the correct password in response to prompting at the operator's console (ATTEMPTS).
- A user-supplied authorization verification routine (AUTHORIZATION).
- That a user-supplied module is to be given control when an exception occurs during alternate-index processing (EXCEPTIONEXIT).

- The owner of the alternate index or component (OWNER).
- A retention period (FOR, TO) and whether the alternate index or data component is to be erased when its entry is deleted (ERASE, NOERASE).
- The share options to be associated with the alternate index or component (SHAREOPTIONS).
- Whether space is to be preformatted before data is initially loaded (RECOVERY, SPEED) and whether write-check operations are to be performed as records are inserted in the alternate index (WRITECHECK, NOWRITECHECK).

DEFINE ALTERNATEINDEX Parameters

ALTERNATEINDEX(options)
An alternate index is to be defined. ALTERNATEINDEX is followed by the parameters specified for the alternate index as a whole; they are optionally followed by the DATA and/or INDEX parameters and their subparameters.

Abbreviation: AIX

ATTEMPTS(number)
The maximum number of times (0 through 7) the operator can try to enter a correct password in response to a prompting message. If 0 is specified, the operator is not prompted and is not allowed to enter a password from the console. This value can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’). This parameter only has an effect when the alternate index is password-protected.

Abbreviation: ATT

Default: 2

AUTHORIZATION(entrypoint)
specifies that, in addition to passwords, you are supplying a USVR (user security verification routine) to check the authority of a processing program to access your alternate index (file). VSE/VSAM transfers control to the USVR only after the program trying to open the file gives a correct password other than the master password. (The USVR is always bypassed whenever a correct master password is specified.) For more information, refer to the "User Security-Verification Routine" topic in the VSE/VSAM User's Guide and Application Programming.

entrypoint - specifies the entry point of the USVR. The entry point may contain one through eight alphameric, national (@, #, and $), or special (hyphen and 12-0 overpunch) characters. The first character must be an alphabetic or a national character.

string - specifies up to 255 characters that are to be passed to the USVR when it receives control to verify authorization. The string must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a string must be coded as two single quotation marks if the string is enclosed in single quotation marks. If the string is specified in hexadecimal, it must be preceded by X and enclosed in single quotation marks (X‘string’).

Abbreviation: AUTH

BLOCKS(primary [secondary])
specifies, for FBA devices only, the number of blocks to be allocated to the alternate index. Every block is 512 bytes long. This parameter can be specified at the alternate index level, the data component level, or at both the data and index component levels. If you specify secondary allocation at the data component level, you should also specify it at the index component level.

(primary [secondary]) - specifies the number of blocks to be made available for primary and secondary allocations. If secondary is specified, space for a component can be expanded to include a maximum of (1) 123 extents or (2) 16 extents per volume for a reusable file. For a component with the UNIQUE attribute, 16 extents are allowed per volume. Primary and secondary can be specified in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form. Space is also allocated on the overflow volume if no secondary allocation has been specified. The overflow volume must be specified in VOLUMES or DEFAULT VOLUMES and must contain space of the correct user class. The first allocation on every overflow volume is the primary value entered. Except for extra-large dataset, the total size of a VSE/VSAM data set cannot exceed 4GB (X'FFFFFFFF' bytes).

Note the following:

- If you specify a primary or secondary value that is not a multiple of a minimum CA, VSE/VSAM rounds it up to a minimum-CA multiple. For example, if you specify BLOCKS(40 20) for generic FBA, VSE/VSAM rounds 40 to 64 and 20 to 64, because minimum CA = 64. For SCSI devices, the minimum CA is 512 blocks. If you specify BLOCKS(40 20) for a SCSI device, VSE/VSAM rounds 40 to 512 and 20 to 512.

Restriction:: The maximum value of the BLOCKS parameter is 16,777,215 (X'FFFFFF'). If the specified value, rounded up to a minimum- or maximum-CA multiple, is larger than 16,777,215, an error message is issued and processing is stopped.

If you specify FREESPACE, VSE/VSAM takes the free space from the requested amount of allocation.

For considerations for UNIQUE files, see the BLOCKS parameter for DEFINE CLUSTER.

TRACKS or CYLINDERS cannot be specified for FBA devices; the RECORDS parameter is valid for FBA devices. For further information on minimum CAs and FBA devices, refer to the "Control Area (CA) Size" in the VSE/VSAM User's Guide and Application Programming.

Abbreviations: BLK or BLOCK

BUFFERSPACE(size)
specifies the minimum space to be provided for buffers. Do not specify BUFFERSPACE at both the alternate index and the data component levels. For more information, refer to the "I/O Buffer Space (Using Non-Shared Resources)" in the VSE/VSAM User's Guide and Application Programming.

size - is the number of bytes to be provided for buffers. This value can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form. The size specified cannot be less than enough space to contain two data component CIs and one index component CI. If you specify an insufficient value, the command terminates.

Abbreviations: BUFSP or BUFSPC

Default: If BUFFERSPACE is omitted, a default value is set in the catalog. This default value, the minimum amount of buffer space allowed by VSE/VSAM, is enough space for two data component CIs and one index component CI.

CATALOG(catname)
identifies the catalog that contains the entry of the base cluster named in the RELATE parameter. You do not have to specify this parameter (unless it is needed for password specification) when the entry is to be defined in the default catalog. (The default catalog is: (1) the job catalog, if one is being used, or (2) the master catalog.) You must define the alternate index in the same catalog as its base cluster.

catname - is the name of the catalog. Make sure the catalog already owns space on the volume (DEFINE SPACE).

password - specifies the catalog's update or higher-level password. If no password is specified and the catalog is password-protected, VSE/VSAM asks the operator for the correct password. If the base cluster's master password is not specified with the RELATE parameter, then the catalog's master password must be specified.

Abbreviation: CAT

CODE(code)
specifies a code name for the alternate index or component. If an attempt is made to access the alternate index without a password, the code name is used in a prompting message to the operator rather than the name of the alternate index. The operator can then provide the correct password.

The code may contain from one through eight EBCDIC characters. The code must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. If a single quotation mark appears within the code, it must be coded as two single quotation marks. Code can also be specified in hexadecimal (X‘code’) form.

If the alternate index or the component is password-protected and CODE is not specified and an attempt is made to access the alternate index or component without supplying a password, the

operator is prompted with the name of the alternate index or component. The operator can then provide the correct password. If the cluster's master password is not specified with the RELATE parameter, then the catalog's master password must be specified.

CONTROLINTERVALSIZE(size)
specifies, in number of bytes, the size of the CIs for the alternate index and/or its components. You can specify this value in decimal (size), hexadecimal (X‘size’), or binary (B‘size’) form.

Note: It is usually undesirable to specify CONTROLINTERVALSIZE at the alternate index level because this size will then apply to both the data and index components. Instead, specify the CI size at the data component level only and let VSE/VSAM choose this size for the index component. Even if you specify an acceptable index CI size, IDCAMS may override it.

The size of a CI for a data component ranges from 512 to 32,768 bytes; you must use a multiple of 512 when the size is between 512 and 8,192 bytes, and a multiple of 2,048 when the size is between 8,192 and 32,768 bytes. In either case, the CI size must be at least ten bytes greater than the average record size. The size of a CI for an index component can be any multiple of 512, up to 8,192 bytes. If you code an improper multiple for size, VSE/VSAM selects the next higher multiple.

If CONTROLINTERVALSIZE is not coded, VSE/VSAM determines the size of CIs as follows: if you specified a value for RECORDSIZE and the size of your data records permits, VSE/VSAM uses 2,048 for the data component size and 512 for the index component size. If you did not specify RECORDSIZE, VSE/VSAM uses 4,096 for the data component size.

For additional information on CIs and block sizes, refer to the "Control Interval (CI) Size" in the VSE/VSAM User's Guide and Application Programming.

Abbreviations: CISZ and CNVSZ

CONTROLPW(password)
specifies a control password for the alternate index or component. The control password permits read and write operations using CI access and all operations permitted by the update and read passwords. For more information, refer to the "Passwords to Authorize Access" in the VSE/VSAM User's Guide and Application Programming.

password - is a one through eight EBCDIC character password. If the password contains commas, semicolons, blanks, parentheses, or slashes, the password must be enclosed in single quotation marks. Single quotation marks within a password must be coded as two single quotation marks if the password is enclosed in single quotation marks. Password may also be coded in hexadecimal form (X‘password’).

Abbreviation: CTLPW

CYLINDERS(primary [secondary])
specifies the number of cylinders to be allocated. Either this parameter or the BLOCKS, TRACKS, or RECORDS parameter must be specified (unless you specified the MODEL parameter) whenever you are defining an alternate index, even when the alternate index has the attribute UNIQUE. Do not specify CYLINDERS or TRACKS for FBA devices or BLOCKS for CKD devices. This parameter can be specified at either the alternate index level, the data component level, or at both the data component and index component levels. If secondary allocation is provided at the data component level, it should also be provided at the index component level.

primary - specifies the number of cylinders available for primary and secondary allocation. The amount(s) can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form. If secondary is specified, space for a component can be expanded to include a maximum of 123 extents or 16 extents per volume for a reusable (dynamic) file. For a component with the UNIQUE attribute, 16 extents are allowed per volume. The total size of a VSE/VSAM data set cannot exceed 4GB (X'FFFFFFFF' bytes).

One primary value is allocated on the first volume when the component is defined. VSE/VSAM makes subsequent allocations on that volume if you specify a secondary value. When there is not enough space of the correct use class for another secondary extent, space is allocated on an overflow volume. Space is also allocated on an overflow volume if no secondary allocation has been specified. The

overflow volume must be specified in VOLUMES or DEFAULTVOLUMES and must contain space of the correct use class. The first allocation on every overflow volume is always the primary value.

If KEYRANGES is specified and the data component is to be contained on several volumes, the primary value is immediately allocated on every volume required for the key ranges.

The secondary value is allocated on all volumes on an as-needed basis regardless of the specification of KEYRANGES.

If you specify FREESPACE, VSE/VSAM takes the free space from the requested amount of allocation.

Abbreviation: CYL

DATA(options)
specifies attributes of the data component of the alternate index; if these attributes are specified for the alternate index as a whole, they are overridden by the options specified in the DATA parameter. If a name is not specified for the data component, a name is generated and listed for it.

DEFAULTVOLUMES(see VOLUMES)

ERASE | NOERASE
specifies whether the data component being defined is to be erased (overwritten with binary zeros) when its entry in the catalog is deleted.

Abbreviations: ERAS and NERAS

Default: NOERASE

Note: ERASE and NOERASE do not function with spanned record files.

EXCEPTIONEXIT(mname)
specifies the name of the user module (i.e., the phase name in the sublibrary) to be given control when an exception occurs during processing of the alternate index or its components. An exception is any condition which causes a SYNAD exit to be taken. The user exception exit is taken before the SYNAD exit, if both are specified.

An abnormal termination occurs if the exception exit is to be loaded but cannot be because it is not in the sublibrary. If it is in the sublibrary but the load operation fails because of insufficient virtual storage, VSE/VSAM ignores the exception exit. The SYNAD exit, if specified, is taken independently.

When the exit receives control, it is in the same AMODE that was in effect when the request was issued.

Abbreviation: EEXT

FATDASD
On ECKD devices, unique datasets (CLUSTER or AIX) must not allocate mixed extents of FAT-DASD and non FAT-DASD. The mix of Small and BIG-DASDs (3390 mod 9/27) (max. 10017 cylinders) is still allowed and not affected.

- In order to define a CLUSTER or AIX as UNIQUE with option FAT-DASD, make sure that each volume used for the assigned extents
  - either is predefined as FAT-DASD to the current VSAM catalog,
  - or has a minimum real capacity of 64K+1 tracks.

If you do not specify FAT-DASD during DEFINE CLUSTER UNIQUE or DEFINE AIX UNIQUE, make sure that no specified volume is already defined as FAT-DASD to the current VSAM catalog. Otherwise the DEFINE command is rejected.

Specified volumes which are unknown to the current VSAM catalog but have a minimum real capacity of 64K+1 tracks, are defined as BIG-DASDs (max 10017 cylinders) if option FAT-DASD has been omitted. These volumes are no longer considered as FAT-DASDs.

Abbreviations: FD and FAT

FILE(dname)
specifies the file name of the DLBL job control statement that, together with an EXTENT statement, identifies the volumes to be used for space allocation. You must specify the FILE parameter only if you specify UNIQUE. The pertinent volumes must be mounted during the define operation.

A FILE parameter must be specified for every unique component, individually or on the alternate index level. If both components of an alternate index are unique and reside on separate volumes, a single FILE parameter may be specified at the alternate index level. This case requires one DLBL statement with EXTENT statements for each of the two components. You must specify allocation information in the EXTENT statement(s) that corresponds to a unique component.

If both components are unique and reside on the same volume, FILE must be specified for both DATA and INDEX. In this case the components cannot share the same DLBL/EXTENT statements. Note that the symbolic unit on the EXTENT statement is not required.

FOR(days) | TO(date)
specifies the retention period for the alternate index. If neither TO nor FOR is specified, the alternate index can be deleted at any time.

FOR(days)
specifies the number of days for which the alternate index is kept. If the number specified is 0 through 1830, the alternate index is retained for the number of days specified; if the number is 1831 through 9999 (maximum), the expiration date of the alternate index is considered a never-expire date. This value can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

TO(date)
specifies the new date until which the alternate index is to be kept. The date has the form yyyyddd, where yyyy (current year through 2099) is the year and ddd (001 through 366) is the day. The new expiration date cannot be a past date and cannot be more than 99 years in the future. The old date form yyddd is also allowed for compatibility reasons. The yy value will be implicitly converted into a yyyy value, so that the date relates to the future. A value of the *yyddd* portion greater than 99365 (such as 99999, 1999999, or 2099999) will be considered to mean "retain indefinitely, does not expire".

FREESPACE(cipercent)
specifies the amount of space that is to be left empty after the initial load or any extension and after any split of CIs (cipercent) or CAs (capercent) during sequential processing. The empty space is available for data records that are subsequently updated and inserted into the file.

The amounts are specified as percentages. The percentages, which must be equal to or less than 100, may be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’).

VSE/VSAMincludes free space as a portion of the storage allocated to you. VSE/VSAM does *not* add the free space amount to your storage request.

There are some cases in which the amount of free space provided is different from the amount requested. In these cases, the value you requested is stored in the catalog. For example:

- If you specify FREESPACE(100 100), the CIs and CAs are not left empty. VSE/VSAM writes one record in the first CI of every CA; the rest of the CIs in the CA are left empty.
- If you specify a *cipercent* that is more than zero but less than the size of the maximum logical record, VSE/VSAM may not reserve enough space in the CI to contain a logical record. If you specify a *capercent* that is more than zero but less than one CI for every CA, VSE/VSAM reserves one CI for every CA.

For additional explanations on free space calculations and examples, refer to the "Distributed Free Space" in the VSE/VSAM User's Guide and Application Programming.

Abbreviation: FSPC

Default: 0 percent

INDEX(options)
specifies attributes of the index component of the alternate index; if these attributes are specified for the alternate index as a whole, they are overridden by the INDEX parameter. If a name is not specified for the index component, a name is generated and listed for the index component.

Abbreviation: IX

KEYRANGES(lowkey highkey)
Portions of the alternate index's data component are to be placed on different volumes. Every portion of the alternate index is called a “key-range”.

The maximum number of key ranges is 123. Key ranges must be in ascending order, and are not allowed to overlap. A gap can exist between two key ranges - you are not allowed to insert records within the gap. For more information, refer to the "Multiple Volume Support" in the VSE/VSAM User's Guide and Application Programming.

The KEYRANGES parameter interacts with other DEFINE ALTERNATEINDEX parameters. You should take care to ensure that, when you specify KEYRANGES, the alternate index's other attributes can be satisfied.

- VOLUMES: There should be as many volume-serial-numbers in the volser list as there are key ranges. When a volume serial number is duplicated in the volser list, several key ranges are allocated space on the volume. When several key ranges are contained on a volume, UNIQUE cannot be coded for the alternate index's data component.

  When there are more volumes in the volser list than there are key ranges, the excess volumes are used for overflow records from any key range without consideration for key-range boundaries.

  When there are fewer volumes in the volser list than there are key ranges, the excess key ranges are allocated on the last volume specified. UNIQUE cannot also be specified.
- UNIQUE: When UNIQUE is specified, every key range resides on its own volume in its own VSE/VSAM data space. Other key ranges for the alternate index cannot also reside on the volume.
- ORDERED: There is a one-for-one correspondence between the volumes in the volser list and the key ranges: the first volume on the volume list contains the first key range, the second volume contains the second key range, and so on. If a volume cannot be allocated in the order specified by the volser list, your alternate-index-definition job terminates with an error message.
- KEYS: The length of the key values must not exceed the key length specified in the KEYS parameter.

The values lowkey and highkey can be 1 to 64 characters long or, if coded in hexadecimal, 1 to 128 characters long. All EBCDIC characters are allowed. Keys consisting of EBCDIC characters must be enclosed in single quotation marks if they contain commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a key must be coded as two single quotation marks if the key is enclosed in single quotation marks. If the key is specified in hexadecimal, it must be coded (X‘key’).

lowkey - specifies the low key of the key range. If lowkey is shorter than the actual keys, it will be padded on the right with binary zeros.

highkey - specifies the high key of the key range. If highkey is shorter than the actual keys, it will be padded on the right with binary ones. KEYRANGES cannot be specified when REUSE is specified.

Abbreviation: KRNG

KEYS(length offset)
specifies information about the alternate key field in the base cluster's data record.

length offset - specifies the length of the key field in bytes (between 1 and 255), and its displacement (offset) from the beginning of the base cluster's data record. The sum of length and offset cannot exceed the length of the shortest record nor the length of the first segment of a spanned record. These values can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

Default: 64 0

MASTERPW(password)
specifies a master password for the alternate index or component. If MASTERPW is not specified and if other passwords exist, the highest-level existing password automatically becomes the master password. The master password allows all operations. See CONTROLPW for the definition of password.

Abbreviation: MRPW

MODEL(entryname)
The catalog entry of an already-defined alternate index is to be used as a model for the entry being built. The already-defined alternate index can have space allocated to it or not. For more information about MODEL, refer to the "Using an Object as a Model" in the VSE/VSAM User's Guide and Application Programming.

entryname - specifies the name of the entry to be used as a model. The entry to be used as a model must be of the same entry type as the entry being built. For example, if you are using this parameter at the data component level, the object used as a model must be a data component.

password - specifies a password. If the alternate index or component whose entry is to be used as a model is password-protected and is cataloged in a password-protected catalog, either the alternate index's or component's password or its catalog's password is required. If both the entry password and the catalog password are specified, the catalog password will be used. If the protection attributes are to be copied, give the master password of either the model (following entryname) or the catalog (following catname). If the model's passwords are not to be copied, any password of either the model or its catalog can be used.

catname - specifies the name of the catalog that contains the entry to be used as a model. This parameter is required if (1) you are going to specify the password of the catalog that defines the alternate index or component instead of specifying the password of the alternate index or component itself, or (2) the catalog is not the default catalog. (The default catalog is (1) the job catalog if one is being used, or (2) the master catalog.)

Restrictions: Your job may be terminated because of allocation problems if you use the MODEL parameter and (1) specify a device type different from that specified for the model through the VOLUMES parameter, (2) change the length of keys through the KEYS parameter, (3) change the size of records, buffer space, or CIs through the BUFFERSPACE, or CONTROLINTERVALSIZE parameter, (4) change from UNIQUE to SUBALLOCATION, or vice versa, or (5) change the unit of allocation through the TRACKS, BLOCKS, CYLINDERS, or RECORDS parameter.

If modeling causes the alternate index or component to be defined as unique, FILE must also be specified.

NAME(entryname)

specifies the name of the alternate index or component. NAME must be specified for the alternate index; it can optionally be specified for a data or index component. The data or index component name must be unique. If no name is specified for a component, an automatic name is generated. The format of the automatic name is the same as for the cluster name and is described under the NAME parameter of the DEFINE CLUSTER command on page “NAME(entryname) ” on page 111.

If the names of the components are not specified, you must give, in a subsequent ALTER command, the system-generated names. Because an alternate index, data component, and index component are individually named, they can be processed individually.

The name may contain from 1 through 44 alphameric characters, national characters (@, #, and $), and special characters (hyphen and blank as 12-0 overpunch). Names containing more than eight characters must be segmented by periods; one through eight characters may be specified between periods. The first character of any name or name segment must be either an alphabetic or a national character. The last character in the name cannot be a period.

NOALLOCATION
indicates that no space allocation is to take place for this define. This allows you to create (a) a dynamic file (alternate index), (b) a model (default list of parameters) that can be used for the future definitions of all alternate indexes, or (c) a model that can be used for the future definition of one

or more specific alternate indexes. For more information on modeling and dynamic files, refer to the "Using an Object as a Model" and "NOALLOCATION" in the VSE/VSAM User's Guide and Application Programming.

- You create a dynamic file (alternate index) by specifying the NOALLOCATION parameter together with REUSE. No space is suballocated to the dynamic file at define time. Rather, the required space (specified with a space allocation parameter at define time) is suballocated to the file when VSE/VSAM opens it.

  When a dynamic file is closed, the file is reset to empty (high-used RBA set to 0, and all space is deallocated) if one of the following is true:

  – The ACB specifies:

  ```
  (a) PARMS=(CLOSDSP=DELETE) or
  (b) PARMS=(CLOSDSP=DATE) and the expiration date has been reached
  ```

  – The DLBL statement specifies:

  ```
  DISP=(,DELETE) or
  DISP=(,DATE) and the expiration date has been reached
  ```

  For information on the second close disposition (which you can specify either in the ACB or in the // DLBL statement), refer to the "CLOSE Disposition"in the VSE/VSAM User's Guide and Application Programming .

- You create a default model for the future definitions of all alternate indexes by specifying in this DEFINE command (a) the NOALLOCATION parameter (b) a reserved entryname (DEFAULT.MODEL.AIX) and (c) the specific list of parameters that you want the model to have. The parameters in this model can be viewed as a system default because they effectively override the actual system default when the parameter is not explicitly specified.

  No data space is ever occupied by this type of model; it exists solely as an entry in a catalog. This also applies to any space allocation parameters you specify with NOALLOCATION. VSE/VSAM uses them for space allocation in subsequent defines.

- You create a model that can be explicitly specified via the MODEL parameter for one or more specific alternate indexes by specifying in the DEFINE command (a) the NOALLOCATION parameter, (b) an ordinary entryname, and (c) the specific parameters to be included in the model.

  This differs from the preceding case because here you can have several models for alternate indexes. You use this type of model by specifying its entryname (instead of a reserved entryname) in the MODEL parameter of a subsequent DEFINE ALTERNATEINDEX command.

In the case of default models, the BUFFERSPACE, RECORDSIZE, CONTROLINTERVALSIZE, RELATE, or the 4 password parameters (READPW, etc.) are not modeled.

If you specify NOALLOCATION for the data component of an alternate index, you must also specify it for the index component. You can (and normally would) specify it only at the alternate index level, in which case it propagates to both the data and index component levels.

Abbreviation: NAL

NOERASE (see ERASE)

NONUNIQUEKEY (see UNIQUEKEY)

NOREUSE (see REUSE)

NOUPGRADE (see UPGRADE)

NOWRITECHECK (see WRITECHECK)

ORDERED | **UNORDERED**
specifies whether volumes are to be used in the order in which they were listed in the VOLUMES parameter. If KEYRANGES is also specified, all of the records within the range specified by the first low-key/high-key pair are placed on the first volume specified in VOLUMES; all of the records within

the second range are placed on the second volume; etc. If it is impossible to allocate volumes in the given order and ORDERED is specified, DEFINE processing is terminated.

Abbreviations: ORD and UNORD

Default: UNORDERED

OWNER(owner ID)
identifies the owner of the alternate index. The owner ID may contain one through eight EBCDIC characters. The owner ID must be enclosed in single quotation marks if it contains a comma, semicolon, blank, parenthesis, or slash. If a single quotation mark appears within owner ID, it must be coded as two single quotation marks when the owner ID is enclosed in single quotation marks. Owner ID may also be coded in hexadecimal form. If specified in hexadecimal, it must be coded (X‘owner ID’).

READPW(password)
specifies a read password for the alternate index or component. The read password permits read operations. See CONTROLPW for the definition of password.

Abbreviation: RDPW

RECORDS(primary [secondary])
specifies the number of records for which space is to be allocated.

Restriction:

In the RECORDS parameter, you can specify a maximum value of 16,777,215 (X'FFFFFF'). If you need to specify a larger allocation, either code a secondary allocation value (to allow for extensions), or use the BLOCKS or CYLINDERS parameters.

If RECORDS is specified at the alternate index level, the space must be large enough to include the index records, or the file will not hold the expected number of data records.

See the CYLINDERS parameter for the information and restrictions that apply to CKD devices. See the BLOCKS parameter for information on FBA disks.

Note: To enable VSE/VSAM to allocate enough space for the number of records you specify, you should let VSE/VSAM choose a CI size for you (see the discussion of the CONTROLINTERVALSIZE parameter). If you choose a CI size such that a whole number of CIs does not fit into a CA, there will be unused space in every CA, and VSE/VSAM might not be able to allocate enough space.

VSE/VSAMassumes that variable length records will be written and, therefore, reserves space for three bytes of control information for every record.

If you specify FREESPACE, VSE/VSAM takes the free space from the requested amount of allocation.

Abbreviation: REC

RECORDSIZE(average maximum)
specifies the average and maximum lengths, in bytes, of the data record of an alternate index. RECORDSIZE (average) must account for five bytes of control information, plus the length of the alternate key, plus the length of at least one base cluster pointer. (The length of a base cluster pointer is the length of the prime key for a key-sequenced file or the length of an RBA (always four bytes) for an entry-sequenced file.) If NONUNIQUEKEY is also used, RECORDSIZE must account for the increased record size resulting from the multiple prime key or RBA pointers in the alternate index data record.

For the format of an alternate index record, see Appendix B, “Format of an Alternate-Index Record,” on page 311.

These values can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’).

The `average` record size must not exceed 32758 bytes (maximum CI size minus ten bytes of CI control information).

The `maximum` record size cannot exceed CA size minus ten bytes of control information for every CI contained in the CA.

Note: The maximum record size influences the partition GETVIS requirements. This is because a sufficiently large storage area to obtain a record of maximum size is allocated for each request.

Abbreviation: RECSZ

Default: 4086 and 32600 are used for the average and maximum record sizes. Default value 4086 overrides the data component CONTROLINTERVALSIZE default value of 2048.

RECOVERY | **SPEED**
specifies whether CAs allocated to the data component are to be preformatted before alternate-index records are loaded into them. The RECOVERY and SPEED parameters apply only to initial loading.

When RECOVERY is specified or defaulted to, the data component's CAs are written with records that indicate end-of-file. When an alternate-index record is written into a CI, it is always followed by a record that identifies the just-written record as the last record in the alternate index. If the initial load fails, your program may be able to resume loading alternate index records after the last correctly written record (because an end-of-file indicator identifies it as the last record; that is, no more alternate index records follow). The BLDINDEX, IMPORT, and REPRO commands do not provide the resume-loading capability.

When SPEED is specified, the data component's space is not preformatted. Its space might contain data records from a previous use of the space, or it might contain binary zeros. Its contents are unpredictable. If the initial load fails, you must load the alternate-index records again from the beginning (because VSE/VSAM is unable to determine where your last correctly written record is, VSE/VSAM can't find a valid end-of-file indicator when it searches your alternate-index records).

When you specify or default to RECOVERY, your initial load takes longer because the CAs are written initially with end-of-file indicators, and again with your alternate-index records. When you specify SPEED, your initial load is quicker.

You cannot use the VERIFY command to recover from a system failure that occurred during the initial load of a file unless you specify or default to RECOVERY.

Abbreviation: RCVY Default: RECOVERY

RELATE(entryname)
names the alternate index's base cluster. The base cluster is an entry-sequenced file or a key-sequenced file that the alternate index is to be organized over. This parameter is required except when you are defining a default model.

entryname - names the base cluster. It must not be a relative record file or a cluster with the REUSE or NOALLOCATION attribute.

password - specifies the base cluster's master password. If the base cluster is password-protected, its master password is required. Alternatively the master password of the catalog in which the base cluster is defined may be specified in the CATALOG parameter.

Abbreviation: REL

REUSE | **NOREUSE**
specifies whether the alternate index can be opened repetitively as a new file, that is, with its high-used RBA set to 0.

When a reusable alternate index is opened for output, its high-used RBA is set to 0 if you open it with (1) an ACB which specifies MACRF=RST or (2) a DLBL statement that specifies DISP=NEW. When you use the IDCAMS BLDINDEX command to rebuild a reusable alternate index, the high-used RBA is always reset to 0 when the alternate index is opened by BLDINDEX.

When a reusable alternate index is closed, its high-used RBA is set to 0 if: (1) the ACB specifies (a) PARMS=(CLOSDSP=DELETE) or (b) PARMS=(CLOSDSP=DATE) and the expiration date has been reached, or (2) the DLBL statement specifies DISP=(,DELETE) or DISP=(,DATE) and the expiration date has been reached. For information on the second close disposition (which you can specify either in the ACB or in the // DLBL statement), refer to the "CLOSE Disposition" in the VSE/VSAM User's Guide and Application Programming.

Reusable files can have several volumes and are restricted to a maximum of 16 physical extents per volume. If a base cluster is defined as reusable, it must not have alternate indexes associated with it; however, it is permissible to define reusable alternate indexes that are related to a nonreusable base cluster.

REUSE must not be specified together with UNIQUE or KEYRANGES.

Abbreviations: RUS and NRUS

Default: NOREUSE

SHAREOPTIONS(value1 value2)
specifies how a file (data or index component of an alternate index) can be shared within one z/VSE system, or across two or more z/VSE systems.

Files that are shared *across z/VSE systems* must reside on a shared disk device.

Regardless of the share option values you specify:

- If a file is currently open for another program, you cannot delete, reset, or alter the share option value of the file.
- During the initial load of a file, VSE/VSAM treats the share option specification as if it were SHAREOPTIONS(1). After the file is loaded and successfully closed, VSE/VSAM uses the specified share option value.

(value1) or *(value1 value2)* specifies the degree of file sharing on one or more z/VSE systems. If a file cannot be shared for the type of sharing you specify, your request to open the file is denied. Every opened ACB counts as one 'request'. The values that can be specified are:

value1=1
Specifies that the file being defined can be opened by any number of requests for input processing.

Once the file is opened for input, it cannot be opened for output processing. Conversely, once the file has been opened for output processing, no other request can open it for input or output until the first output processing has been completed. Share option 1 ensures full read and write integrity.

value1=2
Specifies that the file being defined can be opened by any number of requests for input processing, even if one request is using it for output processing.

Only one request can open the file for output at one time (every ACB counts as one request). Share option 2 ensures write integrity only (because the file might be modified while records are being retrieved from it). Therefore, read integrity of the file must be ensured by every user.

value1=3
Specifies that the file can be opened by any number of requests for both input and output processing.

Except for initial load processing, VSE/VSAM does nothing to ensure read or write integrity. VSE/VSAM, however, ensures that the opened file is not deleted or reset and that its share option value is not altered.

Share option 3 provides good performance, but at the expense of data integrity. The option is intended for users who provide their own read and write integrity.

You should *not* specify this option if the file is accessed by multiple requests and if you do *not* provide your own integrity checking. Too many errors can occur that VSE/VSAM is unable to detect or correct. For example:

User-A opens a file to change some records. VSE/VSAM reads the CI containing the desired records into a buffer so that User-A can update them. The changes, however, are not made in the file itself until after User-A releases the buffer and VSE/VSAM writes the contents of the buffer into the file.

User-B wants to read some records in the same CI (before VSE/VSAM has written the changes made by User-A into the file). Now, User-B actually reads in the unchanged version of the CI. User-B has no way of knowing that some records are already out-of-date, because VSE/VSAM also does not know.

If User-A releases the buffer *before* User-B does, all of the changes made by User-A will be overlaid by the changes made by User-B. There is no way that VSE/VSAM can trace what happened.

Many problems that occur under share option 3 may be avoided if you change to share option 2 or 4. For that reason, if you use share option 3 for data that is being accessed by multiple requests and a problem occurs, you should specify share option 2 or 4, reload the file, and rerun the job.

value1=4 value2≠4

The file can be opened by any number of requests (ACBs) for input processing. At the same time the file can be opened by one or more requests on a single system for output processing. The system that gains exclusive control of the file for output processing will be the one that first issues a request for output processing.

VSE/VSAMensures *write integrity* every time a record is updated or inserted as in share option 2. You can ensure *read integrity* by retrieving records with the *update* option. If you do not use the update option, some records in CIs being updated concurrently by several requests might be missed or skipped by VSE/VSAM, because every request might retrieve a different copy of the CI.

Share option 4 is not valid for an entry-sequenced file. If you specify share option 4, VSE/VSAM treats the specification as though share option 2 had been specified.

If you specify share option 4 for the index component of a file, its data component must also be share option 4.

value1=4 value2=4

The file can be opened by any number of requests (ACBs) for output processing from multiple systems.

VSE/VSAM ensures write integrity every time a record is updated or inserted (as with share option 2). Read integrity of the file must be ensured by every user.

If you specify SHAREOPTIONS(4 4), read integrity is the same as with SHAREOPTIONS(4).

Note: With SHAREOPTIONS(4 4) specified, the lock-file activity (regarding z/VSE DASD sharing) increases. This may have a performance impact.

DFSMSdfp Compatibility: In *value2* you can also specify 3. This value is solely for purposes of compatibility with DFSMSdfp.

Abbreviation: SHR

Default: 1 3

SPEED (see RECOVERY)

SUBALLOCATION (see UNIQUE)

TO (see FOR)

TRACKS(primary [secondary])

specifies the number of tracks to be allocated. If the space is allocated to an alternate index with the UNIQUE attribute, the space is rounded up to the nearest cylinder. If you specify a number equal to or greater than the number of tracks per cylinder, then the allocation is rounded in terms of the number of cylinders needed to contain the specified tracks. See the CYLINDERS parameter for more information and restrictions.

If you specify FREESPACE, VSE/VSAM takes the free space from the requested amount of allocation.

Abbreviation: TRK

UNIQUE | **SUBALLOCATION** | **NOALLOCATION**
specifies whether the alternate index's components are allocated space of their own, whether a portion of previously-defined VSE/VSAM data space is to be used for every component, or whether no space is to be allocated to the alternate index (see NOALLOCATION).

UNIQUE cannot be specified together with REUSE.

If SUBALLOCATION is specified, a data space must exist on the volume on which the alternate index's components are to reside. The name of the data space, not of the component, appears in the VTOC.

If UNIQUE is specified:

- The alternate index's components are allocated space of their own and their names appear in the VTOC of the volume(s) under their own names.
- For the data component, the index component must reside on the same device type and model.
- Every volume specified through the VOLUMES parameter must have at least one and no more than 16 EXTENT statements.
- For CKD devices, data and index component extents must begin on a cylinder boundary.
- For FBA devices, data extents must begin on a minimum CA (track) boundary.
- A FILE parameter must be specified for every unique component individually or on the alternate index level.
- And KEYRANGES is also specified, every key range must be on a different volume and there cannot be fewer volumes specified through the VOLUMES parameter than there are key-range pairs specified through KEYRANGES.

Abbreviations: UNQ, SUBAL, and NAL

Default: SUBALLOCATION

UNIQUEKEY | **NONUNIQUEKEY**
specifies whether the alternate key-value can be found in one or more than one of the base cluster's data records.

UNIQUEKEY indicates that every alternate key is in one and only one data record in the base cluster. When you are building the alternate index via the BLDINDEX command, IDCAMS issues a warning message if multiple records are found with the same alternate key. All but the first multiple record are ignored.

NONUNIQUEKEY indicates that an alternate key can be in several records in the base cluster. If you specify NONUNIQUEKEY, then RECORDSIZE must account for the increased record size resulting from the multiple prime key or RBA pointers in the alternate index data record.

Abbreviations: UNQK and NUNQK

Default NONUNIQUEKEY

UNORDERED (see ORDERED)

UPDATEPW(password)
specifies an update password for the alternate index or component. The update password permits read and write operations. See CONTROLPW for the definition of password.

Abbreviation: UPDPW

UPGRADE | **NOUPGRADE**
specifies whether the alternate index is to be upgraded (that is, kept up to date) when its base cluster is modified. UPGRADE specifies that when the base-cluster's records are added to, updated, or erased, the cluster's alternate index is upgraded to reflect the changed data.

The UPGRADE attribute is not effective for the alternate index until the alternate index is built (see the BLDINDEX command). If the alternate index is defined when the base cluster is open, the UPGRADE attribute takes effect the next time the base cluster is opened.

If you specify UNIQUEKEY and a duplicate key condition is detected during alternate index upgrading, VSE/VSAM issues a return code and backs out any changes already made to other alternate indexes in the upgrade set.

Abbreviations: UPG and NUPG

Default: UPGRADE

USECLASS(primary)
specifies the class of data space to be occupied by a nonunique (suballocated) alternate index or component. For the primary allocation of space:

0
specifies that the alternate index or component is to occupy class 0 data space. Data spaces defined under DFSMSdfp VSAM are class 0 data spaces. 0 is the default space classification.

1-7
The alternate index or component is to occupy the specified class of data space (user-defined).

For the secondary allocation of space:

0
Class 0 data space is to be used by the alternate index or component.

P
The primary space classification is to be used. P is the default.

Whenever the alternate index or component increases in size so as to extend on to additional volumes, the first suballocation on every overflow volume is directed to the class of space specified by the primary subparameter. All subsequent allocations on every overflow volume use the class of space that is specified by the secondary subparameter. If the alternate index or component is extended after you IMPORT CONNECT to z/OS® or VSE/ESA, no USECLASS assignments are made (the alternate index or component is extended into any available data space because class specifications apply to VSE/VSAM only). The alternate index or component is still processable under VSE/VSAM, but the LISTCAT output may be inaccurate.

If you assign a non-zero USECLASS value to a unique alternate index or component, the DEFINE will terminate.

If USECLASS is specified only at the alternate index level, it is applied also to the data and index component levels. If USECLASS is specified at the data level only, it is applied also to the index level. If you want different classifications assigned to the data and index components of an alternate index you must specify (or default) different values at both the data and index levels. In either case, you must specify USECLASS at the same level as you specified space parameters (TRACKS, BLOCKS, CYLINDERS, RECORDS).

Abbreviation: USCL

VOLUMES(volser)|DEFAULTVOLUMES
specifies the volumes to contain the alternate index or components.

VOLUMES
can be specified as a parameter of ALTERNATEINDEX or DATA, or of both DATA and INDEX. If you want the data and index components to reside on different volumes, you must specify the VOLUMES parameter together with both DATA and INDEX. If several volumes are specified with the VOLUMES parameter, they must be of the same device type and capacity (for example, VSE/VSAM will not extend the same level of a file across an IBM 3370 Model 1 and Model 2). Exception: If you are establishing (defining) a default model, you can specify different device types in the model.

You can specify up to 123 volumes for every component. A volume number may contain one to six alphameric, national (@,#, and $) and special characters (commas, blanks, semicolons, parentheses, slashes, asterisks, periods, quotation marks, ampersands, plus signs, hyphens, and equal signs). Single quotation marks within a volume serial number must be coded as two single quotation marks. A volume serial number must be enclosed in single quotation marks if it contains

a special character. For consistency with VSE job control, only alphameric characters should be used.

The volumes to be used can be *virtual disks*, but note that the catalog and the object to be defined must *both* reside on virtual devices.

DEFAULTVOLUMES
indicates that VSE/VSAM is to select the volume(s) it needs from a list of volume(s) established in a default model instead of from volumes specified in the named model (when used in conjunction with the MODEL parameter) or the current define (for more information, refer to the "Default Volumes" in the VSE/VSAM User's Guide and Application Programming). A default model of the correct type is required to be in the relevant catalog.

An error occurs if you specify DEFAULTVOLUMES and VOLUMES together at the same level. You can specify DEFAULTVOLUMES at any level and in any combination of levels; if specified at the alternate index level, DEFAULTVOLUMES propagates to the data and index component levels. If DEFAULTVOLUMES propagates to the data and index component levels where a VOLUMES parameter has been specified, the VOLUMES parameter takes precedence. If DEFAULTVOLUMES is specified (or defaulted to) at any level with the ORDERED or UNIQUE parameters, VSE/VSAM indicates an error.

Although you can specify any number of default volumes, VSE/VSAM will only select up to 16 for a particular component.

Abbreviations: VOL and DFVOL

Default: DEFAULTVOLUMES

WRITECHECK | NOWRITECHECK
specifies whether to check the data transfer of records written in the alternate index. If WRITECHECK is specified, a record is written and then read, without data transfer, to test for the data check condition. If NOWRITECHECK is specified, a record is written, but no checking occurs.

Abbreviations: WCK and NWCK

Default: NOWRITECHECK

DEFINE CLUSTER

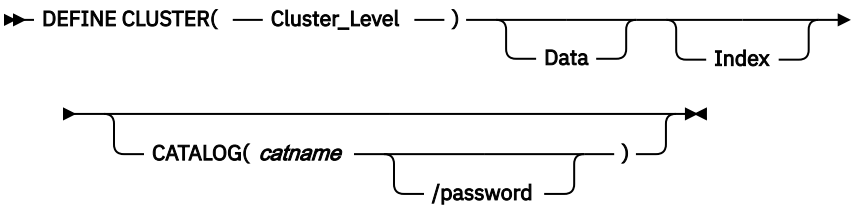
The DEFINE CLUSTER command (DEF CL) is used to define key-sequenced, entry-sequenced, relative-record, or VRDS files. For a key-sequenced or a VRDS file, three entries are created in the catalog: an entry for the cluster, its data component, and its index component. For an entry-sequenced or a relative-record file, two entries are created: one for the cluster and one for its data component. For examples and more information, see:

- “Defining a VSE/VSAM File (Cluster)” on page 22
- “Example 3: Define VSE/VSAM Files” on page 212
- “Example 4: Define NonVSAM and VSE/VSAM Files” on page 215

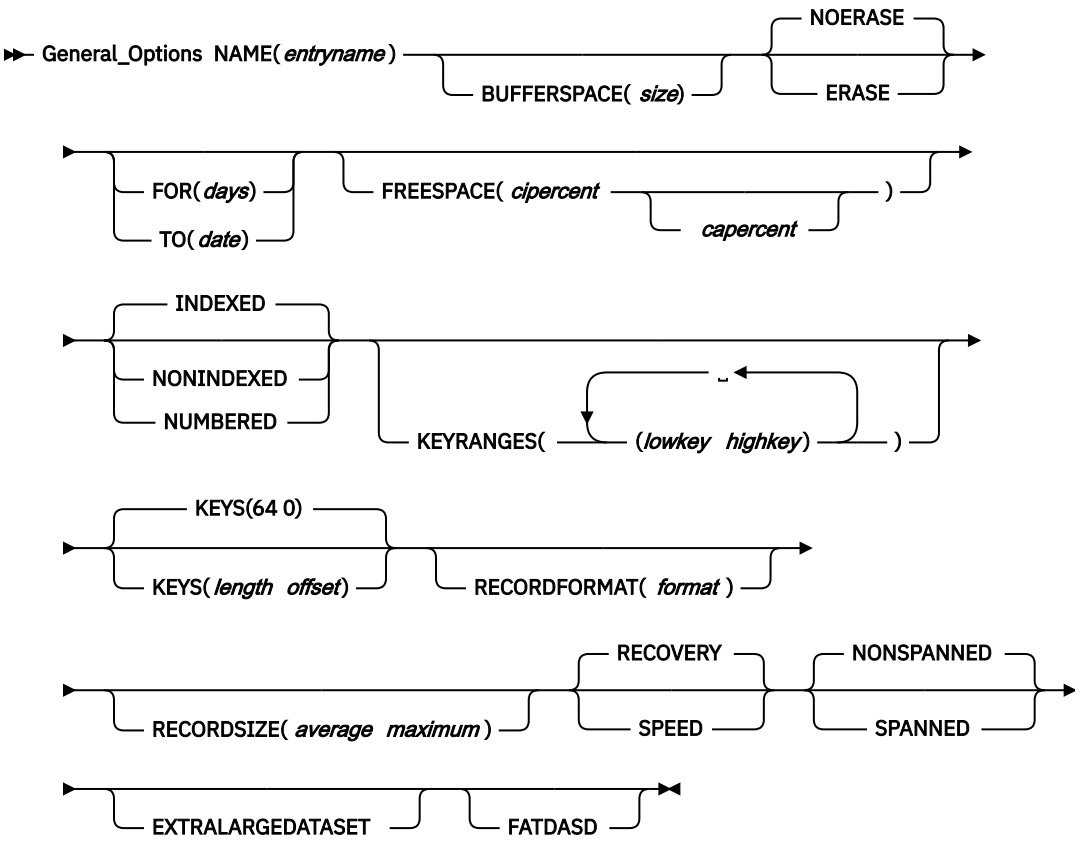
If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (See the description of the SYNCHK parameter of “PARM” on page 205 for further information.)

The format of the DEFINE CLUSTER command is shown below. Note that the organization of the command consists of three groupings. These groupings are referred to as levels in this publication; note that some parameters appear in several levels. (The optional CATALOG parameter stands alone; it does not belong to any level.)

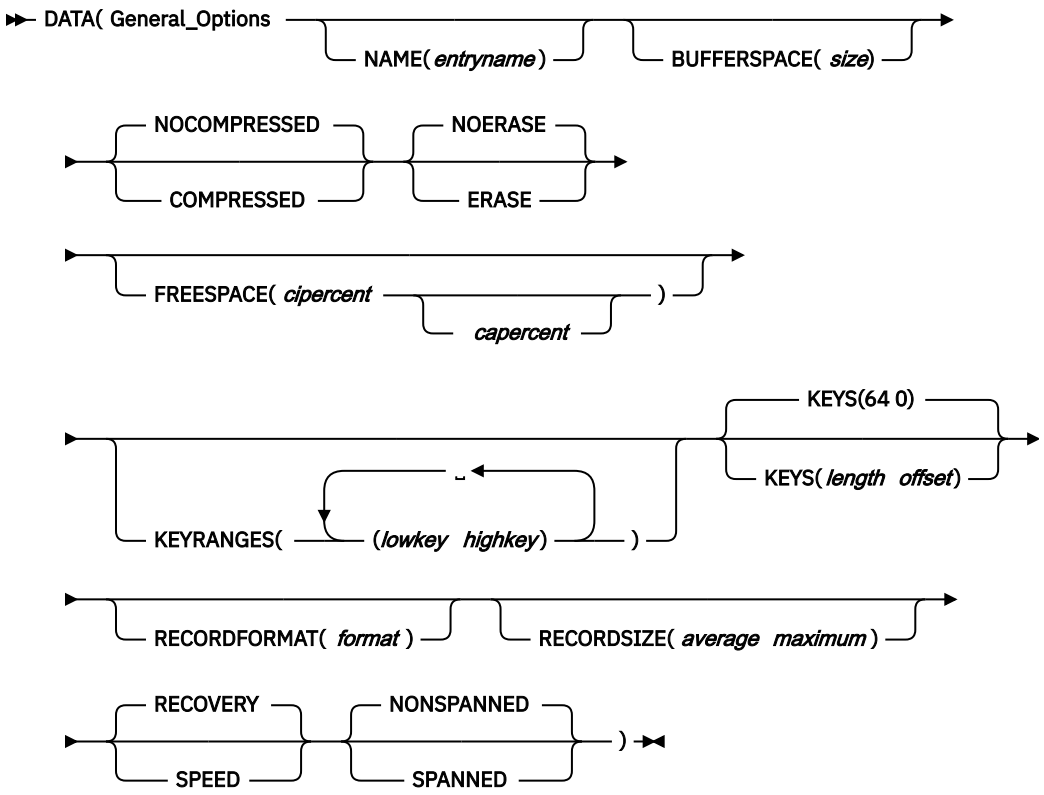
DEFINE CLUSTER



Cluster_Level

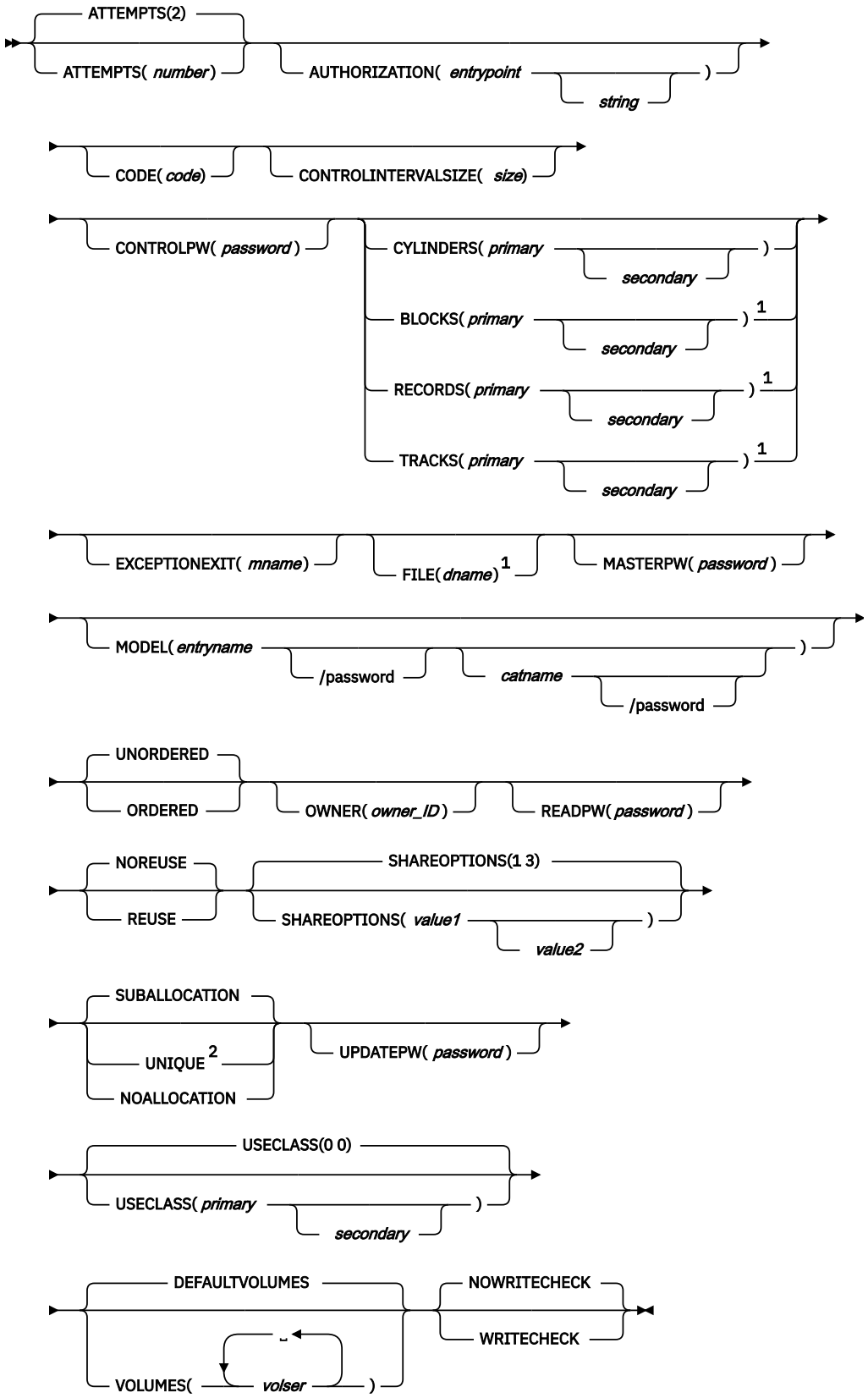


Data



Index

General_Options



Notes:

[1] This parameter is required under certain conditions.
[2] This parameter is invalid under certain conditions.

DEFINE CLUSTER Parameters: Summary

The parameters of the DEF CL command can be divided into these categories: Name, which specifies:

- The name of the cluster (NAME) and, optionally, of its components. This parameter is required.
- The name of the catalog in which the cluster is to be defined (CATALOG).
- The name of the cluster entry to be used as a model for this cluster (MODEL). For more information, refer to the "Using an Object as a Model"in the VSE/VSAM User's Guide and Application Programming .

Data organization, which specifies:

- If the file is NUMBERED and the average and the maximum record length are equal, then it is an RRDS. If average and maximum record length are not equal, it is a VRDS.
- The length and offset of the cluster's key-field (KEYS).
- Whether the cluster is to be reusable (REUSE, NOREUSE).

Allocation, which specifies:

- The volume(s) for space allocation (DEFAULTVOLUMES, VOLUMES). FILE must be specified to provide data space extents only if UNIQUE is specified. VOLUMES or DEFAULTVOLUMES can be specified at the cluster level, data component level, or data and index component level.
- The amount of space to be allocated (TRACKS, CYLINDERS, BLOCKS, or RECORDS). One of these parameters is required (unless MODEL is specified.)
- The particular class of space to be suballocated to the file (USECLASS). For more information, refer to the "Data Space Classification" in the VSE/VSAM User's Guide and Application Programming.
- For an indexed cluster of a type KSDS and VRDS, the percentage of free space to be distributed through the data component of the cluster (FREESPACE).
- Whether a cluster is to reside in a unique or previously-defined data space (UNIQUE, SUBALLOCATION), or have no space allocated to it (NOALLOCATION).
- For an indexed cluster, whether data is to be ranged by key across volumes (KEYRANGES) and, if so, whether the volumes are to be allocated in the order specified (ORDERED, UNORDERED).
- The space to be provided for buffers (BUFFERSPACE) and the size of CIs (CONTROLINTERVALSIZE).
- The average and maximum lengths of data records (RECORDSIZE).
- Records that may be larger than a CI and may span CIs (SPANNED, NONSPANNED).
- The FATDASD parameter, it has to be explicitly specified, otherwise VSAM handles a DASD with more than 64K tracks as BIG-DASD (limited to 10017 cylinders).

Protection and integrity, which specifies:

- The passwords to be associated with the cluster or its components (MASTERPW, CONTROLPW, UPDATEPW, READPW).
- A prompting code (CODE) and the number of attempts allowed to provide the correct password in response to prompting at the operator's console (ATTEMPTS).
- A user-supplied authorization verification routine (AUTHORIZATION).
- The owner of a cluster or its components (OWNER).
- A retention period (FOR, TO) and whether the cluster or data component is to be erased when its entry is deleted (ERASE, NOERASE).
- The share options to be associated with the cluster or its components (SHAREOPTIONS).
- Whether space is to be preformatted before data is initially loaded (RECOVERY, SPEED) and whether write-check operations are to be performed as records are inserted in the data or index components (WRITECHECK, NOWRITECHECK).
- That a user-supplied module is to be given control when an exception occurs during file processing (EXCEPTIONEXIT).

DEFINE CLUSTER Parameters

ATTEMPTS(number)
specifies the maximum number of times (0 through 7) the operator can try to enter a correct password in response to a prompting message. If 0 is specified, the operator is not prompted and is not allowed to enter a password from the console. This value can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

Abbreviation: ATT

Default: 2

AUTHORIZATION(entrypoint[string])
specifies that, in addition to passwords, you are supplying a USVR (user security verification routine) to check the authority of a processing program to access your file. VSE/VSAM transfers control to the USVR only after the program trying to open the file gives a correct password other than the master password. (The USVR is always bypassed when a correct master password is specified.) For more information, refer to the "User Security-Verification Routine" in the VSE/VSAM User's Guide and Application Programming.

entrypoint - specifies the entry point name of the USVR. The entrypoint name may contain one through eight alphameric, national (@, #, $), or special (the hyphen and 12-0 overpunch) characters. The first character must be an alphabetic or a national character.

string - specifies up to 255 characters that are to be passed to the USVR when it receives control to verify authorization. The string must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a string must be coded as two single quotation marks if the string is enclosed in single quotation marks. If the string is specified in hexadecimal, it must be coded (X'string').

Abbreviation: AUTH

BLOCKS(primary [secondary])
specifies, for FBA (and SCSI) devices only, the number of blocks to be allocated to the cluster. Every block is a fixed size of 512 bytes. This parameter can be specified at the cluster level, the data component level, or at both the data and index component levels. If you specify secondary allocation at the data component level, you should also specify it at the index component level.

primary - specifies the number of blocks to be made available for primary and secondary allocation. If secondary is specified, space for a component can be expanded to include a maximum of (1) 123 extents or (2) 16 extents per volume for a reusable file. For a component with the UNIQUE attribute, 16 extents are allowed per volume. Primary and secondary can be specified in decimal (n), hexadecimal (X'n'), or binary (B'n') form. Space is also allocated on the overflow volume if no secondary allocation has been specified. The overflow volume must be specified in VOLUMES or DEFAULT VOLUMES and must contain space of the correct user class. The first allocation on every overflow volume is the primary value entered. Except for extra-large dataset, the total size of a VSE/VSAM data set cannot exceed 4GB (X'FFFFFFFF' bytes).

Note the following:

- If you specify a primary or secondary value that is not a multiple of a minimum CA, VSE/VSAM rounds it up to a minimum-CA multiple. For example, if you specify BLOCKS(40 20) for generic FBA, VSE/VSAM rounds 40 to 64 and 20 to 64, because minimum CA = 64. For SCSI devices, the minimum CA is 512 blocks. If you specify BLOCKS(40 20) for a SCSI device, VSE/VSAM rounds 40 to 512 and 20 to 512.
- If you request more than a maximum CA of storage, VSE/VSAM rounds it up to a maximum-CA multiple. For example, if you specify BLOCKS(970 20) for generic FBA, VSE/VSAM rounds 970 to 1920, because maximum CA = 960. For SCSI devices, if you specify BLOCKS(30970 30970), VSE/VSAM rounds 30970 to 61440, because the maximum CA = 30720. If VSE/VSAM cannot allocate space to include the next multiple of maximum CA, it will attempt to allocate space up to the next multiple of *minimum* CA.

Table 7. Minimum CA and Maximum CA for FBA Devices

FBA Device	Blocks per Min CA	Min CA per Max CA	Blocks per Max CA
Generic Virtual FBA	64	15	960
Generic FBA as VM minidisk	64	15	960
FBA-SCSI	512	60	30720

Restriction: The maximum value of the BLOCKS parameter is 16,777,215 (X'FFFFFF'). If the specified value, rounded up to a minimum- or maximum-CA multiple, is larger than 16,777.215, an error message is issued and processing is stopped.

If you specify FREESPACE, VSE/VSAM takes the free space from the requested amount of allocation.

A further consideration applies if you specify the UNIQUE parameter in the DEFINE CLUSTER command. Data space is acquired concurrently with the DEFINE command via the 'beginning block' and 'number of blocks' specifications in the EXTENT statement. If the beginning block does not coincide with a minimum-CA boundary, VSE/VSAM rounds it up to the next minimum-CA boundary; if the ending block (beginning block plus number of blocks) does not coincide with a minimum-CA boundary, VSE/VSAM will round it down to the previous minimum-CA boundary. Again, if you request more than a maximum CA, VSE/VSAM attempts to round up to a maximum CA boundary.

For example, if you specify 40 for the beginning block and 100 for the number of blocks for generic FBA, VSE/VSAM rounds 40 up to 64 blocks and 139 (40 + 100 - 1 = 139) down to 127 (the beginning block is 64 and the ending block is 127).

For a SCSI device, if you specify 181450 for the beginning block and 600 for the number of blocks, VSE/VSAM rounds 181450 up to 181760 and starts to allocate the space starting at block 181760. The ending block is 183807.

Table 8. Beginning and Ending Blocks for FBA Devices

FBA Device	Specification		Generated	
	JCL //EXTENT card	IDCAMS Parameter	Beginning Block	Ending Block
Generic FBA	`// EXTENT ,FBA001,1,0,80064, 4096`	BLOCKS(40 20)	80064	84159
	`// EXTENT ,FBA001,1,0,80064, 4096`	BLOCKS(100 100)	80064	84159
FBA-SCSI	`// EXTENT ,SCSI00,1,0,141854 ,65536`	BLOCKS(2048 512)	142336	145919
	`// EXTENT ,SCSI00,1,0,141854 ,65536`	BLOCKS(20480 2048)	142336	181247
	`// EXTENT ,SCSI00,1,0,181450 ,65536`	BLOCKS(600 600)	181760	183807

Be aware that in some instances, after VSE/VSAM rounds the values you specified, zero space is the resultant allocation. For example, if you specify 40 as the 'beginning block' value and 24 as the 'number of blocks' value)for generic FBA), VSE/VSAM will round 40 up to 64 as before. However, the value 63 (40 + 24 - 1) for the ending block is less than the beginning block value. Therefore no data space is available for allocation.

In the case of a SCSI device, for example, if you specify 30500 for the beginning block and 200 for the number of blocks, VSE/VSAM will round 30500 up to 30720. However, when VSE/VSAM rounds the calculated ending block value of 30699 (30500 + 200 - 1) down to the closest minimum-CA boundary (30208), this value is less than the beginning block value. Therefore no data space is available for allocation (the beginning block is 30720 and the ending block is 30208). This condition is reported by error messages IDC3025I and IDC3009I and can be avoided by allocating a multiple of minimum CA or maximum CA so that rounding does not take place.

TRACKS or CYLINDERS cannot be specified for FBA devices; the RECORDS parameter is valid for FBA devices. For further information on minimum CAs and FBA devices, refer to the "Control Area (CA) Size" in the VSE/VSAM User's Guide and Application Programming.

Abbreviations: BLK or BLOCK

BUFFERSPACE(size)
specifies the minimum space to be provided for buffers. Do not specify BUFFERSPACE at both the cluster and the data component levels. For more details, refer to the "I/O Buffer Space (Using Non-Shared Resources)" in the VSE/VSAM User's Guide and Application Programming.

size - is the number of bytes to be provided for buffers. This value can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. The size specified cannot be less than enough space to contain two data component CIs and, if the file is key-sequenced, one index CI. If you specify an insufficient value, the command terminates.

For Large DASD, this parameter will not reduce the CA size.

Abbreviations: BUFSP or BUFSPC

Default: If BUFFERSPACE is omitted, a default is set in the catalog. This default value, the minimum amount of buffer space allowed by VSE/VSAM, is enough space for two data component CIs and, if the file is key-sequenced, one index component CI.

CATALOG(catname[password])
identifies the catalog in which the cluster is to be defined. You do not have to specify this parameter (unless it is needed for password specification) when the cluster is to be defined in the default catalog. (The default catalog is (1) the job catalog if one is being used, or (2) the master catalog.)

catname - is the name of the catalog. Make sure the catalog already owns space on the volume (DEFINE SPACE).

password - specifies the update or higher-level password. If no password is specified and the catalog is password-protected, VSE/VSAM asks the operator for the correct password.

Abbreviation: CAT

CLUSTER(options)
specifies that a cluster is to be defined. CLUSTER is followed by the parameters specified for the cluster as a whole. They are optionally followed by the DATA and/or INDEX parameters and their subparameters.

Abbreviation: CL

CODE(code)
specifies a code name for the cluster or component. If an attempt is made to access the cluster without a password, the code name is used in a prompting message to the operator rather than the name of the cluster. The operator can then provide the correct password. The code may contain from one through eight EBCDIC characters. The code must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. If a single quotation mark appears within the code, it must be coded as two single quotation marks. Code may also be specified in hexadecimal (X'code') form.

If the cluster or the component is password-protected and CODE is not specified and an attempt is made to access the cluster or component without supplying a password, the operator is prompted with the name of the cluster or component. The operator can then provide the correct password.

COMPRESSED | NOCOMPRESSED
specifies whether the cluster is to be stored in compressed format. Compressed clusters usually take up less space than noncompressed clusters. Refer to the "Working with Compressed Files" in the VSE/VSAM User's Guide and Application Programming for considerations regarding compressed clusters.

The following types of data sets may be defined with the COMPRESSED attribute:

- KSDS files, i.e. clusters with the INDEXED attribute, under the following condition:
  - The maximum record length must be greater than the following sum: `key_offset + key_length + 40`.
- ESDS files, i.e. clusters with the NONINDEXED attribute, under the following conditions:
  - the maximum record length must be greater than 40
  - the RECORDFORMAT attribute must not be specified (this implies that SAM files in VSAM manages space cannot be compressed).
  - The file is not defined for use as a virtual tape.
- VRDS files, i.e. clusters with the NUMBERED attribute, under the following conditions:
  - The average record size is not equal to the maximum record size.
  - The maximum record length must be greater than 40.

COMPRESSED must not be specified with DEFAULT.MODEL data sets, AIX, RRDS, SAM ESDS, ESDS files defined for use as virtual tapes.

Note: The compression control data set must have been defined to the target catalog previously. Refer to "How to Define the Compression Control Data Set" in the VSE/VSAM User's Guide and Application Programming for more information.

Abbreviations: COMPRESS, CMP, NOCOMPRESS, and NOCMP

CONTROLINTERVALSIZE(size)
specifies, in number of bytes, the size of the CIs for the cluster or its components. You can specify this value in decimal (size), hexadecimal (X‘size’), or binary (B‘size’) form.

It is usually undesirable to specify CONTROLINTERVALSIZE at the cluster level, because this size will then apply to both the data and index components. Instead, specify the CI size at the data component level only and let VSE/VSAM choose this size for the index component. Even if you specify an acceptable index CI size, IDCAMS may override it.

The size of a CI for a data component ranges from 512 to 32,768 bytes; you must use a multiple of 512 when the size is between 512 and 8,192 bytes, and a multiple of 2,048 when the size is between 8,192 and 32,768 bytes. For nonspanned files, the CI size must be at least seven bytes greater than the maximum record size; for spanned files, the CI size must be at least ten bytes greater than the average record size. The size of a CI for an index component can be any multiple of 512 up to 8,192 bytes. If you code an improper multiple for size, VSE/VSAM selects the next higher multiple.

If CONTROLINTERVALSIZE is not coded, VSE/VSAM determines the size of CIs as follows: if you specified a value for RECORDSIZE and the size of your data records permits, VSE/VSAM uses 2,048 for the data component size. If you did not specify RECORDSIZE, VSE/VSAM uses 4096 for the data component size.

A file with a data component block size other than 512, 1,024, 2,048, or 4,096 bytes, or an index component CI size other than 512, 1,024, 2,048, or 4,096 bytes cannot be directly processed by OS/VS; however, file portability between VSE and OS/VS can be retained via the EXPORT/IMPORT commands.

For SCSI devices, the minimum CI size is 1024 if the key length exceeds 38 bytes.

For additional information on CIs and block sizes, refer to the "Control Interval (CI) Size" in the VSE/VSAM User's Guide and Application Programming.

Abbreviations: CISZ and CNVSZ

CONTROLPW(password)
specifies a control password for the cluster or component. The control password permits opening the file for read and write operations using CI access and all operations permitted by the update and read passwords. For more information, refer to the "Passwords to Authorize Access" in the VSE/VSAM User's Guide and Application Programming.

password - is a one through eight EBCDIC character password. If the password contains commas, semicolons, blanks, parentheses, or slashes, the password must be enclosed in single quotation marks. Single quotation marks within a password must be coded as two single quotation marks if the password is enclosed in single quotation marks. Password can also be coded in hexadecimal (X‘password’) form.

Abbreviation: CTLPW

CYLINDERS(primary [secondary])
specifies the number of cylinders to be allocated. Either this parameter or the BLOCKS, RECORDS or TRACKS parameter must be specified (unless you specified the MODEL parameter) whenever you are defining a cluster - even when the cluster has the attribute UNIQUE. Do not specify CYLINDERS or TRACKS for FBA devices or BLOCKS for CKD devices. This parameter can be specified at either the cluster level, the data component level, or at both the data component and index component levels. If secondary allocation is provided at the data component level, it should also be provided at the index component level.

primary - specifies the number of cylinders available for primary and secondary allocation. The amount(s) can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’). If secondary is specified, space for a component can be expanded to include a maximum of 123 extents or 16 extents per volume for a reusable (dynamic) file. For a component with the UNIQUE attribute, 16 extents are allowed per volume. Except for extra-large dataset, the total size of a VSE/VSAM data set cannot exceed 4GB (X'FFFFFFFF' bytes).

One primary value is allocated on the first volume when the component is defined. VSE/VSAM makes subsequent allocations on that volume if you specify a secondary value. When there is not enough space of the correct use class for another secondary extent, space is allocated on an overflow volume. Space is also allocated on an overflow volume if no secondary allocation has been specified. The overflow volume must be specified in VOLUMES or DEFAULTVOLUMES and must contain space of the correct use class. The first allocation on every overflow volume is always the primary value.

If KEYRANGES is specified and the data component is to be contained on several volumes, the primary value is immediately allocated on every volume required for the key ranges.

The secondary value is allocated on all volumes on an as-needed basis regardless of the specification of KEYRANGES.

If you specify FREESPACE, VSE/VSAM takes the free space from the requested amount of allocation.

For Large DASD, allocation in CYLINDERS is recommended.

Abbreviation: CYL

DATA(options)
specifies attributes of the data component of the cluster; if these attributes are specified for the cluster as a whole, they are overridden by the options specified in the DATA parameter. If a name is not specified for the data component, a name is generated and listed for it.

DEFAULTVOLUMES (see VOLUMES)

ERASE | NOERASE
specifies whether the data component being defined is to be erased only for security purposes (overwritten with binary zeros) when its entry in the catalog is deleted.

Abbreviations: ERAS and NERAS

Default: NOERASE

Note: ERASE and NOERASE do not function with spanned record files.

EXCEPTIONEXIT(mname)
specifies the name of the user module (that is, the phase name in the sublibrary) to be given control when an exception occurs during processing of the cluster or its components. An exception is any condition which causes a SYNAD exit to be taken. The user exception exit is taken before the SYNAD exit, if both are specified.

An abnormal termination occurs if the exception exit is to be loaded but cannot be because it is not in the sublibrary. If it is in the sublibrary but the load operation fails because of insufficient virtual storage, VSE/VSAM ignores the exception exit. The SYNAD exit, if specified, is taken independently.

When the exit receives control, it is in the same AMODE that was in effect when the request was issued.

Abbreviation: EEXT

EXTRALARGEDATASET
Defines a VSE/VSAM KSDS file that can exceed 4 GB in size. Keyed access to KSDS files is transparent to existing applications with the exception of:

- Defining the data set with the DEFINE CLUSTER statement
- Interpreting LIST and PRINT outputs, which show CI numbers instead of RBAs.

For such an extra-large dataset, VSE/VSAM will maintain a relative CI number in the 4-byte RBA field. When required, VSE/VSAM will calculate the true RBA from the relative CI number and the CI size. The maximum number of extents supported is 123, thus allowing a maximum KSDS data set size of 289 GB.

Note:

1. Support for extra-large datasets applies to keyed access only.
2. Such a KSDS cannot be one of the following types:
   - Keyranged
   - Unique

   It also cannot be a VSE/VSAM system file like a Catalog or a Compression Control Data Set (CCDS). RBA or CNV access is also not supported.
3. EXTRALARGEDATASET attribute cannot be modelled. To define XXL KSDS, specify this attribute explicitly.
4. For a description of how to create a KSDS file with extended addressing or migrating of an existing KSDS file into a new extra-large dataset, see “Defining an Extra-Large Dataset” on page 28.

Abbreviation: XXL

FATDASD
On ECKD devices, unique datasets (CLUSTER or AIX) must not allocate mixed extents of FAT-DASD and non FAT-DASD. The mix of Small and BIG-DASDs (3390 mod 9/27) (max. 10017 cylinders) is still allowed and not affected.

- In order to define a CLUSTER or AIX as UNIQUE with option FAT-DASD, make sure that each volume used for the assigned extents
  - either is predefined as FAT-DASD to the current VSAM catalog,
  - or has a minimum real capacity of 64K+1 tracks.

If you do not specify FAT-DASD during DEFINE CLUSTER UNIQUE or DEFINE AIX UNIQUE, make sure that no specified volume is already defined as FAT-DASD to the current VSAM catalog. Otherwise the DEFINE command is rejected.

Specified volumes which are unknown to the current VSAM catalog but have a minimum real capacity of 64K+1 tracks, are defined as BIG-DASDs (max 10017 cylinders) if option FAT-DASD has been omitted. These volumes are no longer considered as FAT-DASDs.

Abbreviations: FD, FAT

FILE(dname)
specifies the file name of the DLBL job control statement that, together with an EXTENT statement, identifies the volumes to be used for space allocation. You must specify the FILE parameter only if you specify UNIQUE. The pertinent volumes must be mounted during the DEFINE operation.

A FILE parameter must be specified for every unique component individually or on the cluster level. If both components of a key-sequenced file are unique and reside on separate volumes, a single FILE parameter may be specified at the cluster level. This case requires one DLBL statement with EXTENT statements for each of the two components. You must specify allocation information in the EXTENT statement(s) that corresponds to a unique component. Note that you can omit the symbolic unit from the EXTENT statement.

If both components are unique and reside on the same volume, FILE must be specified for both DATA and INDEX. In this case the components cannot share the same DLBL/EXTENT statements.

FOR(days) | TO(date)
specifies the retention period for the cluster. FOR and TO can be overridden by DELETE PURGE.

FOR(days)
specifies the number of days for which the cluster is to be kept. If the number specified is 0 through 1830, the cluster is retained for the number of days specified; if the number is 1831 through 9999 (maximum), the expiration date of the the cluster is considered a never-expire date. This value can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

TO(date)
specifies the date until which the cluster is to be kept. The date has the form yyyyddd, where yyyy (GE current date through 2099) is the year and ddd (001 through 366) is the day. The new expiration date cannot be a past date and cannot be more than 99 years in the future. The old date form yyddd is also allowed for compatibility reasons. The yy value will be implicitly converted into a yyyy value, so that the date relates to the future. A value of the *yyddd* portion greater than 99365 (such as 99999, 1999999, or 2099999) will be considered to mean "retain indefinitely, does not expire".

Default: If neither TO nor FOR is specified, the cluster can be deleted at any time.

FREESPACE(cipercent)
specifies, for a key-sequenced file, the amount of space that is to be left empty:

- after the initial load
- after any extension
- during sequential processing:

  after any split of CIs (cipercent)
  after any split of CAs (capercent).

Specify the amounts as percentages. They must be equal to or less than 100, and may be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

VSE/VSAM includes free space as a portion of the storage allocated to you. VSE/VSAM does *not* add the free space amount to your storage request.

In some instances, the amount of free space provided differs from the amount you requested. In these cases, the value you requested is stored in the catalog. For example:

- If you specify FREESPACE(100 100), the CIs and CAs are not left empty. VSE/VSAM writes one record for every CI and one CI for every CA; the rest is free space.
- If you specify a cipercent that is more than zero but less than the size of the maximum logical record, VSE/VSAM may not reserve enough space in the CI to contain a record. If you specify a capercent that is more than zero but less than one CI for every CA, VSE/VSAM reserves one CI for every CA.

For additional explanations on free space calculations and examples, refer to the "Distributed Free Spaces" in the VSE/VSAM User's Guide and Application Programming.

FREESPACE can only be specified for KSDS and VRDS. This parameter will be ignored for all other types of clusters.

Abbreviation: FSPC

Default: 0 percent

INDEX(options)
specifies , for a key-sequenced file, attributes of the index component of the cluster; if these attributes are specified for the cluster as a whole, they are overridden by the INDEX parameter. If a name is not specified for the index component, a name is generated and listed for the index component.

Abbreviation: IX

Restriction:
INDEX must not be specified when NUMBERED or NONINDEXED/NONINDEXED is specified at the cluster level.

INDEXED | **NONINDEXED** | **NUMBERED**
specifies that the cluster being defined is for a key-sequenced file (INDEXED), an entry-sequenced file (NONINDEXED), or a relative-record file (NUMBERED). If INDEXED is specified, an index component is automatically defined and cataloged even if the INDEX parameter is not specified.

If you specify NUMBERED, you cannot specify SPANNED. If you specify NUMBERED or NONINDEXED, you cannot specify FREESPACE, INDEX, KEYRANGES, or KEYS. If you specify NUMBERED and RECORDSIZE equal (this means a RRDS cluster will be defined) you cannot specify FREESPACE.

Abbreviations: IXD, NIXD, and NUMD

Default: INDEXED

KEYRANGES(lowkey highkey)
specifies that portions of a key-sequenced file are to be placed on different volumes; every portion is called a key-range.

The maximum number of key ranges is 123. Key ranges must be in ascending order, and are not allowed to overlap. A gap can exist between two key ranges, but you are not allowed to insert records within the gap. For more information, refer to the "Multiple Volume Support" in the VSE/VSAM User's Guide and Application Programming.

The KEYRANGES parameter interacts with other DEFINE CLUSTER parameters. When you specify KEYRANGES, you should take care to ensure that the cluster's other attributes can be satisfied.

- VOLUMES: There should be as many volume-serial-numbers in the volser list as there are key ranges. When a volume number is duplicated in the volser list, several key ranges are allocated space on the volume. When several key ranges are contained on a volume, UNIQUE cannot be coded for the cluster's data component.

  When there are more volumes in the volser list than there are key ranges, the excess volumes are used for overflow records from any key range without consideration for key-range boundaries.

  When there are fewer volumes in the volser list than there are key ranges, the excess key ranges are allocated on the last volume specified. UNIQUE cannot also be specified.
- UNIQUE: When UNIQUE is specified, every key range resides on its own volume in its own VSE/VSAM data space. Other key ranges for the cluster cannot also reside on the volume.
- ORDERED: There is a one-for-one correspondence between the volumes in the volser list and the key ranges: the first volume on the volume list contains the first key range, the second volume contains the second key range, and so on. If a volume cannot be allocated in the order specified by the volser list, your cluster definition job terminates with an error message.
- KEYS: The length of the key values must not exceed the key length specified in the KEYS parameter.

The values lowkey and highkey can be 1 to 64 characters long or, if coded in hexadecimal, 1 to 128 hexadecimal characters long. All EBCDIC characters are allowed. Keys consisting of EBCDIC characters must be enclosed in single quotation marks if they contain commas, semicolons, blanks,

parentheses, or slashes. A single quotation mark within a key must be coded as two single quotation marks if the key is enclosed in single quotation marks. If the key is specified in hexadecimal, it must be coded (X‘key’).

lowkey - specifies the low key of the key range. If lowkey is shorter than the actual keys, it will be padded on the right with binary zeros. However, the low key must be of the same length in all key ranges.

highkey - specifies the high key of the key range. If highkey is shorter than the actual keys, it will be padded on the right with binary ones. However, the high key must be of the same length in all key ranges.

KEYRANGES cannot be specified when REUSE, NUMBERED, or NONINDEXED is specified.

Abbreviation: KRNG

KEYS(length offset)
specifies information about the key field of records in a KSDS.

length offset - specifies the length of the key-field in bytes (between 1 and 255), and its displacement (offset) from the beginning of the data record, in bytes. The sum of the length and offset cannot exceed the length of the shortest record nor the length of the first segment of a spanned record. These values can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form. KEYS cannot be specified when NUMBERED or NONINDEXED is specified.

If the key length of a SCSI data set exceeds 38 bytes, the CI size must be at least 1024. Note that this is true if KEYS is not specified, because the default is 64.

Default: 64 0

The following restrictions regarding key length apply only to clusters defined on BIG-DASD or FAT-DASD.

Table 9. Minimum CI Sizes Depending on Key Length

Key Length in Bytes	**Minimum CI Size**
7 - 35	1024
36 - 55	2048
> 55	4096

MASTERPW(password)
specifies a master password for the cluster or component. The master password allows all operations. If MASTERPW is not specified and if other passwords exist, the highest-level existing password automatically becomes the master password. See CONTROLPW for the definition of password.

Abbreviation: MRPW

MODEL(entryname, password, catname)
specifies that the catalog entry of an already-defined cluster is to be used as a model for the entry being built. The model (already-defined cluster) can have space allocated to it or not. For more information about MODEL, refer to the "Using an Object as a Model" in the VSE/VSAM User's Guide and Application Programming.

entryname - specifies the name of the entry to be used as a model. The entry to be used as a model must be of the same entry type as the entry being built. For example, if you are using this parameter at the data component level, the object used as a model must be a data component.

password - specifies a password. If the cluster or component (whose entry is to be used as a model) is password-protected and is cataloged in a password protected catalog, the password of the cluster, component, or catalog is required. If both the entry password and the catalog password are specified, the catalog password will be used. If the protection attributes are to be copied, give the master password of either the model (following entryname) or the catalog (following catname). If the model's passwords are not to be copied, any password of either the model or its catalog can be used.

catname - specifies the name of the catalog that contains the entry to be used as a model. This parameter is required if (1) you are going to specify the password of the catalog that defines the cluster or component instead of specifying the password of the cluster or component itself, or (2) the catalog is not the default catalog. (The default catalog is (1) the job catalog if one is being used, or (2) the master catalog.)

Restrictions: Your job may be terminated because of allocation problems if you use the MODEL parameter and one or more of the following conditions:

- Specify a device type different from that specified for the model through the VOLUMES parameter.
- Change the length of keys through the KEYS parameter.
- Change the size of records, buffer space, or CIs through the RECORDSIZE/RECORDSIZE, BUFFERSPACE, or CONTROLINTERVALSIZE parameter.
- Change from UNIQUE to SUBALLOCATION, or vice versa.
- Change the unit of allocation through the TRACKS, BLOCKS, CYLINDERS/CYLINDERS, or RECORDS parameter.
- The entry referred to by the model parameter must not be defined with the COMPRESSED attribute.

If modeling causes the cluster or component to be defined as UNIQUE, FILE must also be specified.

NAME(entryname)

specifies the name of the cluster or component. NAME must be specified for the cluster; it can optionally be specified for a data or index component. The name of a data or index component must be unique. If no name is specified for a component, it is generated automatically according to the following rules:

1. If the last qualifier of the cluster name is CLUSTER, it is replaced with DATA for the data component and with INDEX for the index component.

```
Cluster name: SALES.REGION2.CLUSTER
Generated data name = SALES.REGION2.DATA
Generated index name = SALES.REGION2.INDEX
```

2. If the length of the cluster name is less than or equal to 38 characters, then .DATA is appended to the end of the cluster name for the data component and .INDEX is appended for the index component.

```
Cluster name: DEPT64.ASSET.INFO
Generated data name =  DEPT64.ASSET.INFO.DATA
Generated index name = DEPT64.ASSET.INFO.INDEX
```

3. If the length of the cluster name is between 39 and 42 characters inclusive, then .D is appended to the end of the cluster name for the data component and .I is appended for the index component.

```
Cluster name: DEPTABCD.RESOURCE.REGION66.DATA1234.STUFF
Generated data name = DEPTABCD.RESOURCE.REGION66.DATA1234.STUFF.D
Generated index name = DEPTABCD.RESOURCE.REGION66.DATA1234.STUFF.I
```

4. If the name is longer than 42 characters and the last qualifier is not CLUSTER, the five-qualifier name is produced. The first four qualifiers are taken from the cluster name. The fifth qualifier is started with .D for the data component and with .I for the index component, .D or .I is followed by bits 28-55 of the TOD clock value in the character (printed) form.

```
Cluster name: DIV012.GROUP16.DEPT98.DAILYLOG.DEC1988.BACK
Generated data name = DIV012.GROUP16.DEPT98.DAILYLOG.D99EFB7B
Generated index name = DIV012.GROUP16.DEPT98.DAILYLOG.I1A12FAE
```

If the names of the components are not specified, you must give, in a subsequent ALTER command, the system-generated names. Because a cluster, data component, and (for a key-sequenced file) index component are individually named, they can be processed individually.

You can specify NAME(entryname) to:

1. Simply identify (name) the cluster or component (this is the most common use). You specify a name that contains from 1 through 44 alphameric characters, national characters (@, #, and $), and two special characters (the hyphen and the 12-0 overpunch). Names containing more than eight characters must be segmented by periods; one through eight characters may be specified between periods. The first character of any name or name segment must be either an alphabetic or a national character. The last character in the name cannot be a period.
2. To predefine the work files for every partition in which you anticipate to use them (called partition independence). To take advantage of partition independence, you specify entryname as either of the following:
   - name.pid

     where

     name
     contains a maximum of 27 characters with the same rules as under 1.

     pid
     is one of the following identifiers: BG, F1, F2, F3, F4, F5, F6, F7, F8, F9, FA, FB.
   - %name

     where

     % (percent sign)
     indicates that you want VSE/VSAM to append a partition identifier to the name you specify. VSE/VSAM assigns the ID of the partition in which the DEFINE job is running. For example, if you specify %MY.FILE, VSE/VSAM generates the following: MY.FILE.BG (this assumes that the job was run in the BG partition). To reference this file from another partition, you must specify MY.FILE.BG.

     If you do not want the work file to occupy data space after it is closed, specify the NOALLOCATION parameter in this DEFINE. Refer to the "CLOSE Disposition" in the VSE/VSAM User's Guide and Application Programming.

     name
     contains a maximum of 27 characters with the same rules as described under 1.
3. Indicate that work file space is to be shared between central processors with the capability to run the same job in any partition and any processor without conflict (called partition and central processor independence). To take advantage of partition and central processor independence, you specify entryname as either of the following:
   - name.pid.*.cpu-id.model

     where

     name (27 characters) and pid
     have the same meanings as above.

     ***.cpu-id**
     is the six-character CPU identification number (you must prefix it with a C).

     model
     is the four-character model number (you must prefix it with an M).
   - %%name

     where

     %%
     indicates that you want VSE/VSAM to append a unique processor identification and partition identifier to the name you specify. VSE/VSAM assigns IDs corresponding to the partition and processor in which the DEFINE job is running. For example, if you specify %%MY.FILE, VSE/VSAM generates the following:

MY.FILE.BG.C010658.M4341 (this assumes that the job was run in the BG partition on a processor that had an identification number of 010658 and a model number of 4341). To reference this file from another partition or processor, specify MY.FILE.BG.C010658.M4341.

If you do not want the work file to occupy data space after it is closed, specify the NOALLOCATION parameter in this DEFINE.

name (27 characters)
has the same meaning as above.

NOALLOCATION
indicates that no space allocation is to take place for this define. This allows you to create (a) a dynamic file, (b) a model (default list of parameters) that can be applied, in subsequent define operations, to all the files of a particular file organization, or (c) a model that can be applied, in subsequent define operations, to a specific file. For NOALLOCATION files, VSE/VSAM does not check at define time whether there is sufficient space available. For more information on modeling and dynamic files, refer to the "Using an Object as a Model" and "NOALLOCATION" in the VSE/VSAM User's Guide and Application Programming.

- You create a dynamic file by specifying the NOALLOCATION parameter together with REUSE. No space is suballocated to a dynamic file at define time, rather, the required space (specified with a space allocation parameter at define time) is suballocated to the file when VSE/VSAM opens it. Dynamic files can be entry-sequenced, key-sequenced, or relative-record files.

  When a dynamic file is closed, the file is reset to empty (high-used RBA set to 0, and all space is deallocated) if (1) the ACB specifies (a) PARMS=(CLOSDSP=DELETE) or (b) PARMS=(CLOSDSP=DATE) and the expiration date has been reached, or (2) the DLBL statement specifies DISP=(,DELETE) or DISP=(,DATE) and the expiration date is reached. For information on the second close disposition (which you can specify either in the ACB or in the // DLBL statement), refer to the "CLOSE Disposition" in the VSE/VSAM User's Guide and Application Programming.
- You create a default model for the files of a particular file organization (key-sequenced, entry-sequenced, relative record) by specifying in this DEFINE command the NOALLOCATION parameter, a reserved entry name (see below), and the specific list of parameters that you want the model to have. The parameters in this model can be viewed as a system default because they effectively override the actual system default when the parameter is not explicitly specified.

  No data space is ever occupied by this type of model; it exists solely as an entry in the catalog. This also applies to any space allocation parameters you specify with NOALLOCATION; VSE/VSAM uses them for space allocation in subsequent defines. Every catalog can contain one model for every one of the five different file organizations. The reserved entry names that you specify for a particular file organization are:

  Reserved Entry Name
  File Organization

  DEFAULT.MODEL.KSDS
  Key-sequenced data set

  DEFAULT.MODEL.ESDS
  Entry-sequenced data set

  DEFAULT.MODEL.RRDS
  Relative-record data set

  DEFAULT.MODEL.VRDS
  Variable-length relative-record data set

  DEFAULT.MODEL.AIX
  Alternate index

  DEFAULT.MODEL.ESDS.SAM
  SAM ESDS file

- You create a model that applies to one specific file (and occupies no data space) by specifying in the DEFINE command the NOALLOCATION parameter, an ordinary entry name, and the specific parameters to be included in the model.

  This differs from the preceding case because here you can have several models for a particular file organization. You use this type of model by specifying its entry name (instead of a reserved entry name) in the MODEL parameter of a subsequent DEFINE command.

In the case of default models, the BUFFERSPACE, RECORDSIZE, CONTROLINTERVALSIZE, and the four password parameters (READPW, etc.) are not modeled.

If you specify NOALLOCATION for the data component of a key-sequenced file you must also specify it for the index component. You can (and normally would) specify it only at the cluster level, in which case it propagates to both the data and index component levels.

Abbreviation: NAL

NOERASE (see ERASE)

NONINDEXED (see INDEXED)

NONSPANNED (see SPANNED)

NOREUSE (see REUSE)

NOWRITECHECK (see WRITECHECK)

NUMBERED (see INDEXED)

ORDERED | **UNORDERED**
specifies whether volumes are to be used in the order in which they were listed in the VOLUMES parameter. If KEYRANGES is also specified, all of the records within the range specified by the first low-key/high-key pair are placed on the first volume specified in VOLUMES; all of the records within the second range are placed on the second volume; etc. If it is impossible to allocate volumes in the given order and ORDERED is specified, DEFINE processing is terminated.

Abbreviations: ORD and UNORD

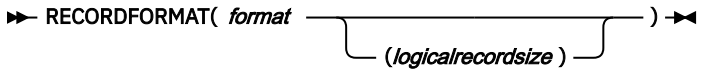
Default: UNORDERED

OWNER(owner ID)
identifies the owner of the cluster. The owner ID may contain one through eight EBCDIC characters. The owner ID must be enclosed in single quotation marks if it contains a comma, semicolon, blank, parenthesis, or slash. If a single quotation mark appears within owner ID, it must be coded as two single quotation marks when the owner ID is enclosed in single quotation marks. Owner ID may also be coded in hexadecimal form (X‘owner ID’).

READPW(password)
specifies a read password for the cluster or component. The read password permits read operations. See CONTROLPW for the definition of password.

Abbreviation: RDPW

RECORDFORMAT



indicates a NONINDEXED file is a SAM ESDS file. Substitute one of the following for format: F, FB (logical recordsize), V, VB, U, or NCIF. For more information refer to the "Explicit Define Cluster (Using the DEFINE CLUSTER Command)" in the VSE/VSAM User's Guide and Application Programming.

Restriction: RECORDFORMAT parameter cannot be specified together with RECOVERY parameter. This applies whether RECOVERY is specified on cluster level or on data level.

Abbreviation: RECFM

RECORDS(primary [secondary])
specifies the number of records for which space is to be allocated.

Restriction:
In the RECORDS parameter, you can specify a maximum value of 16,777,215 (X'FFFFFF')

If RECORDS is specified at the cluster level, the space must be large enough to include the index records or the file will not hold the expected number of data records.

If you specify FREESPACE, VSE/VSAM also takes the free space from the requested amount of allocation. See the CYLINDERS parameter for the information and restrictions that apply to CKD devices. See the BLOCKS parameter for information on FBA devices.

Note: To enable VSE/VSAM to allocate enough space for the number of records you specify, you should let VSE/VSAM choose a CI size for you (see the discussion of the CONTROLINTERVALSIZE parameter). If you choose a CI size such that a whole number of CIs does not fit into a CA, there will be unused space in every CA and VSE/VSAM might not be able to allocate enough space.

VSE/VSAM assumes that variable length records will be written and therefore reserves space for three bytes of control information for every record.

Abbreviation: REC

RECORDSIZE(average maximum)
specifies the average and maximum lengths, in bytes, of the data records (all records are assumed to be of variable length). These values can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form. The minimum record size that can be specified is one. For non-spanned records, the maximum record size cannot exceed CI size minus 7 bytes of control information.

The maximum CI size is 32,768 bytes. If necessary, VSE/VSAM adjusts the CI size to accommodate the maximum record size plus 7 (nonspanned records) or the average record size plus 10 (spanned records).

For spanned records, the maximum record size cannot exceed CA size minus 10 bytes of control information for every CI in the CA. The average record size cannot exceed 32,758 bytes (maximum CI size minus 10 bytes of control information).

When defining a RRDS, the average and maximum record sizes must be equal. If the average and maximum record size of NUMBERED dataset do not match, the cluster will be defined as a variable record data set (VRDS). The maximum record size for VRDS is 32,757 (each record is prefixed with a 4-byte relative record number).

Note that the records of COMPRESSED dataset are prefixed with 3 bytes of compression control information (5 bytes for spanned records). The prefix decreases the maximum record size of COMPRESSED dataset accordingly.

Abbreviation: RECSZ

Default: The default values 4,089 and 4,089 are used for the average and maximum record sizes if SPANNED is not specified, 4,086 and 32,600 if SPANNED is specified. This default (4,086 or 4,089) overrides the data component default of 2,048.

RECOVERY | **SPEED**
specifies whether CAs allocated to the data component are to be preformatted before records are inserted into them. SPEED/RECOVERY applies only to initial loading.

When RECOVERY is specified or defaulted to, the data component's CAs are written with records that indicate end-of-file. When a data record is written (during the initial load) into a CI, it is always followed by a record that identifies the just-written record as the last record in the cluster (Software end-of-file, SEOF). If the initial load fails, your program may be able to resume loading data records after the last correctly-written data record (because an end-of-file indicator identifies it as the last record; that is, no more data records follow). The BLDINDEX, IMPORT, and REPRO commands do not provide the resume-loading capability.

When SPEED is specified, the data component's space is not preformatted. Its space might contain data records from a previous use of the space, or it might contain binary zeros; its contents are unpredictable. If the initial load fails, you must load the data records again from the beginning

(because VSE/VSAM is unable to determine where your last correctly written record is, VSE/VSAM can't find a valid end-of-file indicator when it searches your data records).

When you specify or default to RECOVERY, your initial load takes longer because the CAs are written initially with end-of-file indicators, and again with your data records. When you specify SPEED, your initial load is quicker.

You cannot use the VERIFY command to recover from a system failure that occurred during the initial load of a file unless you specify or default to RECOVERY.

Restriction: RECORDFORMAT parameter cannot be specified together with RECOVERY parameter. This applies whether RECOVERY is specified on cluster level or on data level.

Abbreviation: RCVY

Default: RECOVERY

REUSE | **NOREUSE**
specifies whether the cluster can be opened repetitively as a new file, that is, with its high-used RBA set to 0. This parameter allows the use of a VSE/VSAM file as a work file. When a reusable file is opened for output, its high-used RBA is set to 0 if you open it with

(1) an ACB that specifies MACRF=RST, or
(2) a DLBL statement that specifies DISP=NEW.

When a reusable file is closed, its high-used RBA is set to 0 if:

1. The ACB specifies (a) PARMS=(CLOSDSP=DELETE) or (b) PARMS=(CLOSDSP=DATE) and the expiration date has been reached,or
2. The DLBL statement specifies DISP=(,DELETE) or DISP=(,DATE) and the expiration date has been reached.

For information on the second close disposition (which you can specify either in the ACB or in the // DLBL statement), refer to the "CLOSE Disposition" in the VSE/VSAM User's Guide and Application Programming.

You can define entry-sequenced, key-sequenced, extra-large dataset, and relative-record files as reusable. Alternate indexes cannot be defined over a reusable cluster; however, you can define reusable alternate indexes that are related to a nonreusable base cluster.

Reusable files can have several volumes and are restricted to a maximum of 16 physical extents per volume. REUSE must not be specified together with UNIQUE or KEYRANGES.

Abbreviations: RUS and NRUS

Default: NOREUSE

SHAREOPTIONS(value1 value2)
specifies how a file (data or index component of a cluster) can be shared within one z/VSE system, or across two or more z/VSE systems.

Files that are shared *across z/VSE systems* must reside on a shared disk device.

Regardless of the share option values you specify:

- If a file is currently open for another program you cannot delete, reset, or alter the share option value of the file.
- During the initial load of a file, VSE/VSAM treats the share option specification as if it were SHAREOPTIONS(1). After the file is loaded and successfully closed, VSE/VSAM uses the specified share option value.

(value1) or *(value1 value2)* specifies the degree of file sharing on one or more z/VSE systems. If a file cannot be shared for the type of sharing you specify, your request to open the file is denied. Every opened ACB counts as one 'request'. The values that can be specified are:

value1=1
Specifies that the file being defined can be opened by any number of requests for input processing.

Once the file is opened for input, it cannot be opened for output processing. Conversely, once the file has been opened for output processing, no other request can open it for input or output until the first output processing has been completed. Share option 1 ensures full read and write integrity. The default value is 1.

value1=2
Specifies that the file being defined can be opened by any number of requests for input processing, even if one request is using it for output processing.

Only one request can open the file for output at one time (every ACB counts as one request). Share option 2 ensures write integrity only, because the file might be modified while records are being retrieved from it. Therefore, read integrity of the file must be ensured by every user.

value1=3
Specifies that the file can be opened by any number of requests (ACBs) for both input and output processing.

Except for initial load processing, VSE/VSAM does nothing to ensure read or write integrity. VSE/VSAM, however, ensures that the opened file is not deleted or reset, and that its share option value is not altered.

Share option 3 provides good performance, but at the expense of data integrity. The option is intended for users who provide their own read and write integrity.

You should *not* specify this option if the file is accessed by multiple requests and if you do *not* provide your own integrity checking. Too many errors can occur that VSE/VSAM is unable to detect or correct. For example:

User-A opens a file to change some records. VSE/VSAM reads the CI containing the desired records into a buffer so that User-A can update them. The changes, however, are not made in the file itself until after User-A releases the buffer and VSE/VSAM writes the contents of the buffer into the file.
User-B wants to read some records in the same CI (before VSE/VSAM has written the changes made by User-A into the file). Now, User-B actually reads in the unchanged version of the CI; User-B has no way of knowing that some records are already out-of-date, because VSE/VSAM also does not know.
If User-A releases the buffer *before* User-B does, all of the changes made by User-A will be overlaid by the changes made by User-B. There is no way that VSE/VSAM can trace what happened.

Many problems that occur under share option 3 may be avoided if you change to share option 2 or 4. For that reason, if you use share option 3 for data that is being accessed by multiple requests and a problem occurs, you should specify share option 2 or 4, reload the file, and rerun the job.

value1=4 value2≠4
The file can be opened by any number of requests (ACBs) for input processing. At the same time the file can be opened by one or more requests on a single system for output processing. The system that gains exclusive control of the file for output processing will be the one that first issues a request for output processing.

VSE/VSAMensures *write integrity* every time a record is updated or inserted as in share option 2. You can ensure *read integrity* by retrieving records with the *update* option. If you do not use the update option, some records in CIs being updated concurrently by several requests might be missed or skipped by VSE/VSAM, because every request might retrieve a different copy of the CI.

Share option 4 is not valid for an entry-sequenced file. If you specify share option 4, VSE/VSAM treats the specification as though share option 2 had been specified.

If you specify share option 4 for the index component of a file, its data component must also be share option 4.

value1=4 value2=4
The file can be opened by any number of requests (ACBs) for output processing from multiple systems.

VSE/VSAM ensures write integrity every time a record is updated or inserted (as with share option 2). Read integrity of the file must be ensured by every user.

If you specify SHAREOPTIONS(4 4), read integrity is the same as with SHAREOPTIONS(4).

Note: With SHAREOPTIONS(4 4) specified, the lock-file activity (regarding z/VSE DASD sharing) increases. This may have a performance impact.

VSE/VSAM ignores value2 unless value1=4 and value2=4.

Abbreviation: SHR

Default: 1 3

SPANNED | NONSPANNED
specifies whether or not the maximum length of a data record may be greater than the CI size, that is, whether a data record may span CIs. A spanned record always begins on a CI boundary. All records in a spanned file are not necessarily spanned; multiple records can exist in a single CI in a spanned file.

SPANNED cannot be specified when NUMBERED is specified. Note that spanned records cannot be accessed by programs that use RPL OPTCD=LOC.

If NONSPANNED is specified and the computed CI size cannot be made larger than the maximum record size, an error message is issued and the object is not defined.

Abbreviations: SPND and NSPND

Default: NONSPANNED

SPEED (see RECOVERY)

SUBALLOCATION (see UNIQUE)

TO (see FOR)

TRACKS(primary [secondary])
specifies the number of tracks to be allocated.

If you specify a number equal to or greater than the number of tracks per cylinder, then the allocation is made in terms of cylinders.

If the space is allocated to a cluster with the UNIQUE attribute, the space is rounded up to the nearest cylinder. The allocation for a unique cluster is always made on cylinder boundary. See the CYLINDERS parameter for more information and restrictions.

If you specify FREESPACE, VSE/VSAM takes the free space from the requested amount of allocation.

For Large DASD, allocation in CYLINDERS is recommended.

Abbreviation: TRK

UNIQUE | SUBALLOCATION | NOALLOCATION
specifies whether the cluster's components are allocated space of their own, whether a portion of previously-defined VSE/VSAM data space is to be used for every component, or whether no space is to be allocated to the cluster (see NOALLOCATION).

UNIQUE cannot be specified together with REUSE.

If SUBALLOCATION is specified: a data space must exist on the volume on which the cluster or components are to reside. The name of the data space, not of the component, appears in the VTOC.

If UNIQUE is specified:

- A cluster's data and (for a key-sequenced file) index component is allocated space of its own and the name of the component appears in the VTOC of the volume(s).

- For the data component of a key-sequenced cluster, the index component must reside on the same device type and model.
- For any cluster or component, every volume specified through the VOLUMES/DEFAULT VOLUMES parameter must have at least one and no more than 16 EXTENT statements. The associated logical unit must not be assigned to IGNORE or UA (unassigned).
- For CKD devices, data extents must begin on a cylinder boundary.
- For FBA devices, data extents must begin on a minimum CA (track) boundary.
- A FILE parameter must be specified for every unique component individually or on the cluster level.
- For a key-sequenced cluster with KEYRANGES specified, every key range must be on a different volume and there cannot be fewer volumes specified through the VOLUMES parameter than there are key-range pairs specified through KEYRANGES.
- For a VRDS only the data component can be unique. The internal index needs suballocated space.

Abbreviations: UNQ, SUBAL, and NAL

Default: SUBALLOCATION

UNORDERED (see ORDERED)

UPDATEPW(password)
specifies an update password for the cluster or component. The update password permits read and write operations. See CONTROLPW for the definition of password.

Abbreviation: UPDPW

USECLASS(primary[secondary])
specifies the class of data space to be occupied by a nonunique (suballocated) cluster or component. For the primary allocation of space:

0
specifies that the cluster or component is to occupy class 0 data space. The default space classification is 0. (Data spaces defined under DFSMSdfp VSAM are treated as class 0.)

1-7
specifies that the cluster or component is to occupy the specified class of data space (user-defined).

Class 1 means high performance and is specifically suggested for fixed-head areas.

For the secondary allocation of space:

0
specifies class 0 data space is to be used by the cluster or component.

P
specifies that the primary space classification is to be used. The default is P.

Whenever the cluster or component increases in size so as to extend on to additional volumes, the first suballocation on every overflow volume is directed to the class of space specified by the primary subparameter. All subsequent allocations on every overflow volume use the class of space that is specified by the secondary subparameter. If the file is extended after you IMPORT CONNECT to z/OS or VSE/ESA, no USECLASS assignments are made. (The file is extended into any available data space because class specifications apply to VSE/VSAM only.) The file is still processable under VSE/VSAM, but the LISTCAT output may be inaccurate.

If you assign a nonzero USECLASS value to a unique cluster or component, the DEFINE will terminate.

If USECLASS is specified only at the cluster level only, it is applied also to the data and index component levels. If USECLASS is specified at the data level only, it is applied also to the index level. If you want different classifications assigned to the data and index components of a key-sequenced file you must specify (or default) different values at both the data and index levels. In either case, you must specify USECLASS at the same level as you specified space parameters (TRACKS, BLOCKS, CYLINDERS, RECORDS).

For more information, refer to the "Data Space Classification" in the VSE/VSAM User's Guide and Application Programming.

Abbreviation: USCL

VOLUMES(volser)|DEFAULTVOLUMES
specifies the volume(s) to contain the cluster or component.

VOLUMES
can be specified as a parameter of CLUSTER or DATA, or of both DATA and INDEX. If several volumes are specified with the VOLUMES parameter, they must be of the same device type and model, otherwise unpredictable results and/or data corruption can occur. The only exception is when you are establishing (defining) a default model, then you can specify different device types in the model.

If you want the data and index components to reside on different volumes, you must specify the VOLUMES parameter together with DATA and INDEX.

You can specify up to 123 volumes for every component. A volume serial number, volser, may contain one through six alphameric, national (@, #, and $), and special characters (commas, blanks, semicolons, parentheses, slashes, asterisks, periods, quotation marks, ampersands, plus signs, hyphens, and equal signs). Single quotation marks within a volume number must be coded as two single quotation marks. A volume number must be enclosed in single quotation marks if it contains a special character. For consistency with VSE job control, only alphameric characters should be used.

The volumes to be used can be *virtual disks*, but note that the catalog and the object to be defined must *both* reside on virtual devices.

DEFAULTVOLUMES
indicates that VSE/VSAM is to select the volume(s) it needs from a list of volume(s) established in a default model instead of from volumes specified in the named model (when used in conjunction with the MODEL parameter) or the current define. (For more information, refer to the "Default Volumes" in the VSE/VSAM User's Guide and Application Programming.)

An error occurs if you specify DEFAULTVOLUMES and VOLUMES together at the same level. You can specify DEFAULTVOLUMES at any level and in any combination of levels; if specified at the cluster level, DEFAULTVOLUMES propagates to the data component level and, if applicable, to the index component level. If DEFAULTVOLUMES propagates to the data and index component levels where a VOLUMES parameter has been specified, the VOLUMES parameter takes precedence.

If DEFAULTVOLUMES is specified (or defaulted to) at any level with the ORDERED or UNIQUE parameters, VSE/VSAM indicates an error.

Although you can specify any number of default volumes, VSE/VSAM will only select up to 16 for a particular component.

Abbreviations: VOL and DFVOL

Default: DEFAULTVOLUMES

WRITECHECK | NOWRITECHECK
specifies whether to check the data transfer of records written in the cluster. If WRITECHECK is specified, a record is written and then read, without data transfer, to test for the data check condition. If NOWRITECHECK is specified, a record is written but no checking occurs.

Abbreviations: WCK and NWCK

Default: NOWRITECHECK

DEFINE MASTERCATALOG

The DEFINE MASTERCATALOG command (DEF MCAT) is used to define the master catalog. When you define a master catalog, a data space to contain the catalog is automatically created. Entries for both the master catalog itself and the volume containing the data space are placed in the master catalog. A

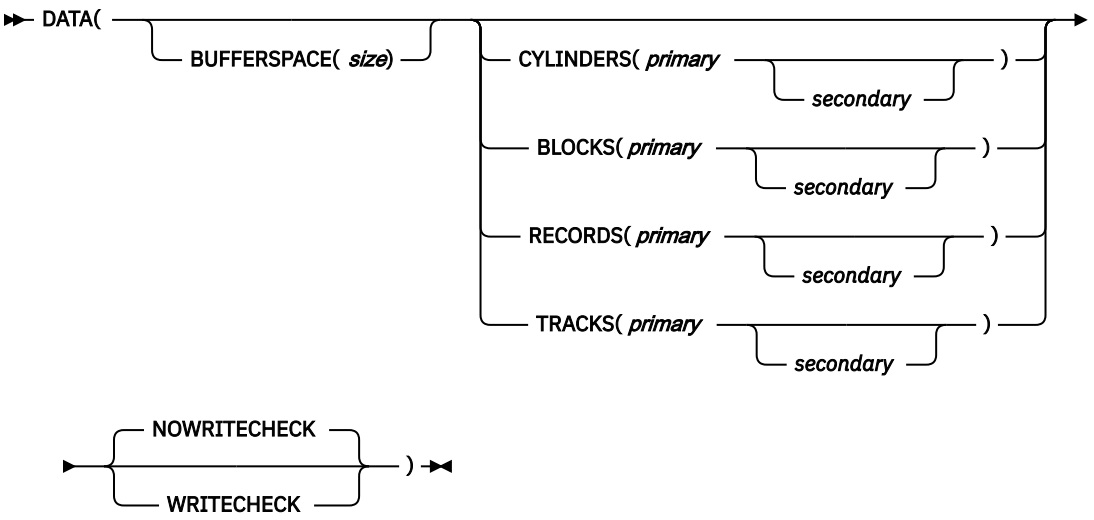
catalog must have a master password if any VSE/VSAM files cataloged in it are to be password-protected. A catalog must have an update or higher-level password if any nonVSAM files cataloged in it are to be password-protected. For more information, see

- "The Master Catalog" on page 16
- "Example 1: Define a System's Catalogs" on page 209

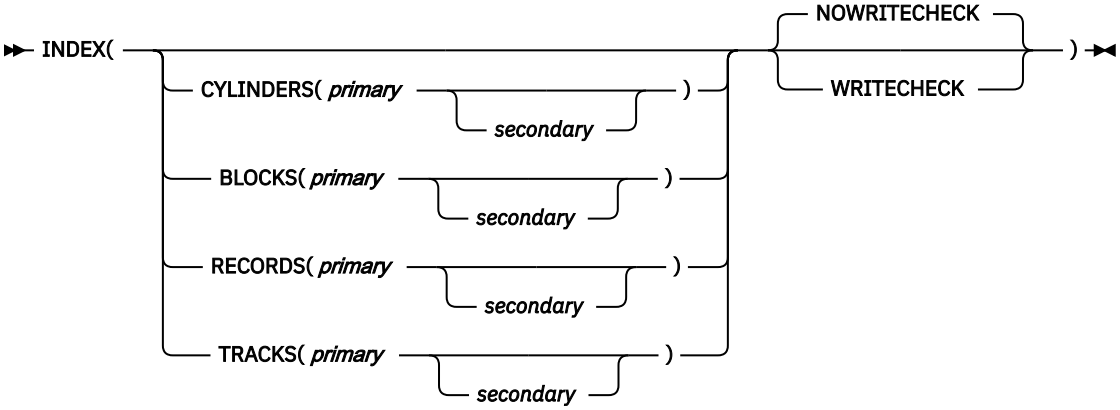
If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under "PARM" on page 205.)

The format of the DEFINE command is shown below. Note that the organization of the command consists of three groupings. These groupings are referred to as levels in this publication. Note that some parameters appear in several levels.

Data



Index



Notes:

[1] DEDICATE and ORIGIN are mutually exclusive.

DEFINE MASTERCATALOG Parameters: Summary

The parameters of the DEF MCAT command can be divided into the following categories:

Name, which specifies:

- The name of the catalog being defined (NAME). This is a required parameter.

Allocation, which specifies:

- The volume and location on the volume where the master catalog is to reside (VOLUME, ORIGIN).
- The amount of space to be allocated (BLOCKS, CYLINDERS, DEDICATE, RECORDS, TRACKS). One of these parameters is required.
- The classification of the catalog's data space (CLASS). For more information, refer to the "Data Space Classification" in the VSE/VSAM User's Guide and Application Programming.
- The space to be provided for buffers (BUFFERSPACE).

Protection and integrity, which specifies:

- Passwords to be associated with the catalog (MASTERPW, CONTROLPW, UPDATEPW, READPW).
- A prompting code (CODE) and number of attempts allowed to provide the correct password in response to prompting at the operator's console (ATTEMPTS).
- A user-supplied security verification routine (AUTHORIZATION).

- The owner of the catalog (OWNER).
- A retention period (FOR, TO).
- Whether write-check operations are to be performed as records are inserted in the catalog (WRITECHECK, NOWRITECHECK).
- The FATDASD parameter, it has to be explicitly specified, otherwise VSAM handles a DASD with more than 64K tracks as BIG-DASD (limited to 10017 cylinders).

DEFINE MASTERCATALOG Parameters

ATTEMPTS(number)
specifies the maximum number of times (0 through 7) the operator can try to enter a correct password in response to a prompting message. If 0 is specified, the operator is not prompted and is not allowed to enter a password from the console. This value can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

Abbreviation: ATT

Default: 2

AUTHORIZATION(entrypoint)
specifies that in addition to passwords, you are supplying a routine to check the authority of a processing program to access the master catalog. VSE/VSAM transfers control to the verification routine only after the program trying to open the master catalog gives a correct password other than the catalog's master password. (The verification routine is always bypassed whenever a correct master password is specified.) For more information, refer to the "User Security-Verification Routine" in the VSE/VSAM User's Guide and Application Programming.

entrypoint - specifies the entry point name of the user security verification routine. The entry point name may contain one through eight alphameric, national (@, #, and $), or special (hyphen and 12-0 overpunch) characters. The first character must be an alphabetic or a national character. Entrypoint is the phase name in the sublibrary.

string - specifies up to 255 characters that are to be passed to the user security verification routine when it receives control to verify authorization. The string must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a string must be coded as two single quotation marks if the string is enclosed in single quotation marks. If the string is specified in hexadecimal, it must be preceded by X and enclosed in single quotation marks (X‘string’).

Abbreviation: AUTH

BLOCKS(primary [secondary])
specifies, for FBA devices only, the number of blocks to be allocated to the master catalog. Each block has 512 bytes. This parameter must be specified at the catalog level. In addition, it can be specified at the data component level, or at both the data and index component levels.

primary - specifies the number of blocks to be made available for primary and secondary allocation. Primary and secondary can be specified in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

The data component requires at least three CAs of primary allocation.

If you specify a primary or secondary value that is not a multiple of a minimum CA, VSE/VSAM rounds it to a minimum-CA multiple.

Note the following:

- If you specify a primary or secondary value that is not a multiple of a minimum CA, VSE/VSAM rounds it up to a minimum-CA multiple. For example, if you specify BLOCKS(40 20) for generic FBA, VSE/VSAM rounds 40 to 64 and 20 to 64, because minimum CA = 64. For SCSI devices, the minimum CA is 512 blocks. If you specify BLOCKS(40 20) for a SCSI device, VSE/VSAM rounds 40 to 512 and 20 to 512.

- To enable VSE/VSAM to allocate enough space for the number of records you specify, you should let VSE/VSAM choose a CI size for you (see the discussion of the CONTROLINTERVALSIZE parameter). If you choose a CI size such that a whole number of CIs does not fit into a CA, there will be unused space in each CA, and VSE/VSAM might not be able to allocate enough space. VSE/VSAM assumes that variable length records will be written and therefore reserves space for three bytes of control information for each record.

Restriction: The maximum value of the BLOCKS parameter is 8,388,096 (X'7FFFFF'). If the specified value, rounded up to a minimum- or maximum-CA multiple, is larger than 8,388,096, an error message is issued and processing is stopped.

A further consideration applies to the BLOCKS specification at the catalog level only. VSE/VSAM acquires the data space for the catalog concurrently with the DEFINE MASTERCATALOG command. If the beginning block number (specified in the ORIGIN parameter) does not coincide with a minimum CA boundary, VSE/VSAM rounds it up to the next minimum CA boundary; if the ending block (ORIGIN value plus BLOCKS value) does not coincide with a minimum CA boundary, VSE/VSAM rounds it down to the previous minimum CA boundary. (Again, if you request more than a maximum CA, VSE/VSAM rounds to a maximum CA boundary.)

For example, ORIGIN (40) and BLOCKS (100) for generic FBA indicate that space for the catalog begins at block number 64 (40 is rounded to 64) and ends at block number 127 (40 + 100 - 1 = 139 is rounded down to 127).

Be aware that in some instances, after VSE/VSAM rounds the values you specified, zero space is the resultant allocation. For example, if you specify ORIGIN (40) and BLOCKS (50) for generic FBA, VSE/VSAM rounds 40 up to 64 as before. However when VSE/VSAM rounds the ending block value of 89 (40 + 50 - 1 = 89) down to the closest minimum CA boundary, the value (63) is less than the beginning block value. Therefore, no data space is available for allocation to the catalog (the beginning block is 64 and the ending block is 63).

Secondary allocation of space to the catalog depends on the following:

- If you do not specify a secondary value, VSE/VSAM itself extends the catalog's data and index components an additional 15 times (one CA at a time if needed).
- If you specify secondary at the data component level, VSE/VSAM uses the specified value to extend (if needed) your catalog an additional 15 times. Note that if the value you specify does not coincide with a CA boundary, VSE/VSAM rounds it down to the nearest multiple (CA boundary) and then uses this multiple for the secondary extensions.
- If you specify a secondary value at the catalog level or index component level, VSE/VSAM ignores it (in this case the specified value is solely an entry in the catalog for purposes of z/OS compatibility) and instead extends the index component by one CA at a time for an additional 15 times if needed.

For SCSI devices, the allocation is rounded up to a 512-block boundary; the minimum primary allocation is 3072 blocks. For FBA devices, it is recommended not to use the VSAM default settings but rather to explicitly define VSAM catalogs with larger primary and secondary allocation sizes. The default catalog on an FBA device will have 256 blocks as primary and 128 blocks as secondary allocation. These values will result in a catalog-full condition after defining 512 files (including the catalog itself). The catalog has already reached 16 extents (the maximum for VSAM catalogs) and cannot be extended further.

TRACKS or CYLINDERS cannot be specified for FBA devices; the RECORDS or DEDICATE parameters are valid for FBA devices. For further information on minimum CAs and FBA devices, refer to "Control Area (CA) Size" in the VSE/VSAM User's Guide and Application Programming. For more information on space allocation, refer to “How Data Space is Assigned to a Catalog” on page 16.

Abbreviations: BLK or BLOCK

BUFFERSPACE(size)
specifies the minimum space to be provided for buffers when the user catalog is used.

size - is the number of bytes to be provided for buffers when the user catalog is used. The minumum amount must be 3072, the maximum value depends on GETVIS available. This value can

be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. 8192 is no longer a limitation, the maximum is calculated in correspondence with available GETVIS 31-bit

Default: 3072 if BUFFERSPACE is not coded.

Abbreviations: BUFSPC or BUFSP

CLASS(value)
specifies the class to be assigned to the total amount of data space allocated for the catalog (even that portion of space not used, if any, by the catalog). The values you can specify (only at the master catalog level) are:

0
Data spaces defined under DFSMSdfp VSAM are treated as class 0.

1-7
specifies that the defined data space is classified according to your own criteria. You can use any criteria to classify the data space; for example, you can assign particular files to the middle section of a volume if you feel that by doing this the performance of your installation will be enhanced.

For more information, refer to "Data Space Classification" in the VSE/VSAM User's Guide and Application Programming.

CODE(code)
specifies a code name for the master catalog. If an attempt is made to access a password-protected catalog without a password, the code name is used in a prompting message to the operator rather than the name of the catalog. The operator can then provide the correct password. The code may contain from one through eight EBCDIC characters.

The code must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. If a single quotation mark appears within the code, it must be coded as two single quotation marks. Code can also be expressed in hexadecimal (X'code') form.

If CODE is not specified and an attempt is made to access a password-protected catalog without supplying a password, the operator is prompted with the name of the master catalog.

CONTROLPW(password)
specifies a control password for the master catalog. The control password permits read and write operations using CI access and all operations permitted by the update and read passwords. Though the master catalog is a form of cluster and conventional clusters may be opened with a control password, the master catalog cannot be opened as a file unless the user supplies its master password. A master password permits such operations as REPRO to be performed on the catalog. For more information, refer to "Passwords to Authorize Access"in the VSE/VSAM User's Guide and Application Programming .

password - is a one through eight EBCDIC character password. If the password contains commas, semicolons, blanks, parentheses, or slashes, the password must be enclosed in single quotation marks. Single quotation marks within a password must be coded as two single quotation marks if the password is enclosed in single quotation marks. Password can also be expressed in hexadecimal (X'password') form.

Note: Authorization checking for files cataloged in the master catalog is performed only if the master catalog is password-protected. If a master catalog has no password protection, but a user catalog does, VSE/VSAM checks passwords for all files defined in that user catalog.

Abbreviation: CTLPW

CYLINDERS(primary [secondary])
specifies, for CKD devices only, the number of cylinders to be allocated. Either this parameter or the BLOCKS, DEDICATE, RECORDS, or TRACKS parameter must be specified at the master catalog level. In addition, they (except DEDICATE) can be specified at the data level, or at both the data and index levels.

See "How Data Space is Assigned to a Catalog" on page 16 for more information.

Do not specify CYLINDERS or TRACKS for FBA devices or BLOCKS for CKD devices. See ORIGIN parameter for boundary requirements.

primary - specifies the number of cylinders available for primary and secondary allocation. These values can be expressed in decimal (n), hexadecimal (X‘n’) or binary (B‘n’) form.

The primary suballocation for the combined data and index components must be contained in a single extent (contiguous data space). The data component requires space for at least three CAs.

Secondary allocation of space to the catalog depends on the following:

- If you specify secondary at the data component level, VSE/VSAM uses the specified value to extend (if needed) your catalog an additional 15 times. Note that if the value you specify does not coincide with a CA boundary, VSE/VSAM rounds it down to the nearest multiple (CA) and then uses this multiple for the secondary extensions.
- If you specify a secondary value at the catalog level or index component level, VSE/VSAM ignores it (in this case the specified value is solely an entry in the catalog for z/OS compatibility) and instead extends the index component by one CA at a time for an additional 15 times if needed.
- If you do not specify secondary value, VSE/VSAM itself extends the catalog's data and index components an additional 15 times (one CA at a time if needed).

It is recommended to use this parameter for Large DASD; the minimum value is 5 cylinders for BIG- and for FAT-DASD. The maximum value for primary and secondary allocation is 5000 cylinders.

Abbreviation: CYL

DATA(options)
specifies attributes of the data component of the catalog. The suballocation of space for the data and index components of the master catalog is discussed in “How Data Space is Assigned to a Catalog” on page 16.

If DATA allocation parameters are not coded, the space specified at the master catalog level is available to contain catalog entries only.

The total amount of space specified through DATA and INDEX parameters cannot be greater than that specified at the master catalog level of the DEFINE MASTERCATALOG command.

DEDICATE
specifies that up to 16 extents of unowned and unallocated space on VOLUME(volser) is to be owned by the master catalog. You must delete expired files from the VTOC before VSE/VSAM will allocate that space on the VTOC. DEDICATE is mutually exclusive with the space allocation parameters (BLOCKS, RECORDS, TRACKS, CYLINDERS) and can be specified at the catalog level only. Do not specify the ORIGIN parameter with DEDICATE. There must be a contiguous area of space on the volume large enough to contain the primary allocation.

You can specify:

- DEDICATE alone - no space parameters (BLOCKS, TRACKS, RECORDS, or CYLINDERS) are specified at the data and index component levels; in this case the data space suballocated to the catalog itself is calculated by VSE/VSAM to be the minimum size catalog and the remaining space is available for later use by VSE/VSAM.
- DEDICATE at the catalog level and a space parameter value at both the data component and index component levels. In this case, the data space suballocated to the catalog itself is the sum of the values specified at the data and index component levels. Any VSE/VSAM data space that is not suballocated at this time is available for later use by VSE/VSAM.
- DEDICATE at the catalog level and a space parameter value at the data component level; in this case VSE/VSAM calculates the space required for the index component and adds this amount to the data component specification. This sum becomes the amount of data space that is suballocated to the catalog. Any VSE/VSAM space that is not suballocated at this time is available for later use by VSE/VSAM.

DEFINE MASTERCATALOG with option DEDICATE on a BIG- or FAT-DASD (ECKD >= 65535 tracks) will be rejected, if previously existing VTOC entries would cause more than one VSAM space extent to be generated.

Abbreviation: DED

FATDASD
FATDASD enables VSAM to allocate up to 65520 cylinders on the specified volume. This definition cannot be changed until the associated catalog is deleted and redefined. The VSAM space bitmap is managed in cylinders instead of tracks.

Specified volumes which are unknown to the current VSAM catalog but have a minimum real capacity of 64K tracks, are defined as BIG-DASDs (max 10017 cylinders) if parameter FATDASD has been omitted.

Abbreviations: FD and FAT

FOR(days) | **TO(date)**
specifies the retention period for the master catalog.

FOR(days)
specifies the number of days the master catalog is to be kept. If the number specified is 0 through 1830, the catalog is retained for the number of days specified; if the number is 1831 through 9999 (maximum), the expiration date of the catalog is as a never-expire date. This value can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

TO(date)
specifies the date until which the master catalog is to be kept. The date has the form yyyyddd, where yyyy (GE current date through 2099) is the year and ddd (001 through 366) is the day. The new expiration date cannot be a past date and cannot be more than 99 years to the future. The old date form yyddd is also allowed for compatibility reasons. The yy value will be implicitly converted into a yyyy value, so that the date relates to the future. A value of the *yyddd* portion greater than 99365 (such as 99999, 1999999, or 2099999) will be considered to mean "retain indefinitely, does not expire".

Default: The expiration date is set to 0, which means that the master catalog can be deleted whenever it is empty.

INDEX(options)
specifies attributes of the index component of the catalog. The suballocation of space for the data and index components of the master catalog is discussed in "How Data Space is Assigned to a Catalog" on page 16.

An allocation parameter (BLOCKS, CYLINDERS, RECORDS, or TRACKS) must have been specified as a parameter of DATA for an allocation parameter to be specified as a parameter of INDEX. The total amount of space specified through the DATA and INDEX subparameters cannot be greater than that specified at the MASTERCATALOG level. The WRITECHECK/ NOWRITECHECK parameter defaults to whatever was specified at the master catalog level.

The primary index allocation will be at least 4% of the primary data allocation, and the secondary index allocation will be the same as the primary allocation.

Abbreviation: IX

MASTERCATALOG(options)
specifies that a master catalog is to be defined. MASTERCATALOG is followed by the parameters specified for the catalog as a whole; they are optionally followed by the DATA and/or INDEX parameters and their subparameters.

Abbreviations: MCAT or MRCAT

MASTERPW(password)
specifies a master password for the master catalog. The AUTHORIZATION, CODE, and ATTEMPTS parameters have no effect unless the catalog is master password-protected. If MASTERPW is not specified and if other passwords exist, the highest-level existing password automatically becomes the

master password. The master password allows all operations; it is required to open the catalog as a file. See CONTROLPW for the definition of password.

Abbreviation: MRPW

NAME(entry name)
specifies the name of the master catalog; it is identical to the file-ID in the IJSYSCT DLBL statement. The name may contain from 1 through 44 alphameric characters, national characters (@, #, and $), and special characters (the hyphen and the 12-0 overpunch). Names containing more than eight characters must be segmented by periods; one through eight characters may be specified between periods. The first character of any name or name segment must be either an alphabetic or national character. The last character in the name cannot be a period. This parameter is required.

NOWRITECHECK (see WRITECHECK)

ORIGIN(tracknumber | blocknumber)
specifies the beginning point (track number or block number) of the master catalog's data space. (If the block number does not coincide with a minimum CA boundary, VSE/VSAM rounds it up to the next minimum CA boundary. See the BLOCKS parameter for a description of roundup.) VSE/VSAM determines the ending point of the data space by adding the value you specify (at the master catalog level) for CYLINDERS, BLOCKS, TRACKS, or RECORDS (VSE/VSAM converts records and blocks to minimum CAs) to the ORIGIN specification. See "Using the ORIGIN Parameter for a Catalog" on page 18 for more information.

Specify ORIGIN at the catalog level only. Specify a value greater than 0 for tracknumber and a value greater than 1 for blocknumber.

You can omit the ORIGIN (and DEDICATE) parameter; in this case the beginning point of the catalog's data space is the first available area on the volume that is large enough to contain your primary allocation (if you specified CYLINDERS, the beginning boundary is a cylinder boundary).

If ORIGIN is specified for a SCSI device, the primary allocation must be explicitly specified with at least 3072 blocks.

Abbreviation: ORG

OWNER(owner ID)
identifies the owner of the master catalog. The owner ID may contain one through eight EBCDIC characters. The owner ID must be enclosed in single quotation marks if it contains a comma, semicolon, blank, parenthesis, or slash. If a single quotation mark appears within owner ID, it must be coded as two single quotation marks when the owner ID is enclosed in single quotation marks. Owner ID can also be expressed in hexadecimal form (thus: 'owner ID').

READPW(password)
specifies a read password for the master catalog. The read password permits read operations. See CONTROLPW for the definition of password.

Abbreviation: RDPW

RECORDS(primary [secondary])
specifies the number of records for which space is to be allocated.

Restriction:

In the RECORDS parameter, you can specify a maximum value of 16,777,215 (X'FFFFFF').

Every record in the catalog contains 512 bytes. See the CYLINDERS parameter for the information and restrictions that apply to CKD devices. See the BLOCKS parameter for information on FBA devices.

Abbreviation: REC

TO (see FOR)

TRACKS(primary [secondary])
specifies the number of tracks to be allocated. Do not specify TRACKS for an FBA device. See the CYLINDERS parameter for more information and restrictions.

For Large DASD, allocation in CYLINDERS is recommended.

Abbreviation: TRK

UPDATEPW(password)
specifies an update password for the master catalog. The update password permits read and write operations. See CONTROLPW for the definition of password.

Abbreviation: UPDPW

VOLUME(volser)
specifies the volume that is to contain the master catalog. No other catalog may reside on this volume. The volume number (volser) may contain one to six alphameric, national (@, #, and $), and special characters (commas, blanks, semicolons, parentheses, slashes, asterisk, periods, quotation marks, ampersands, plus signs, hyphens, and equal signs). Single quotation marks within the volume number must be coded as two single quotation marks. The volume number must be enclosed in single quotation marks if it contains a special character. This parameter is required.

For consistency with Job Control, only alphameric characters should be used.

Note: A master catalog must **not** be defined on a *virtual disk*.

Abbreviation: VOL

WRITECHECK | NOWRITECHECK
specifies whether to check the data transfer of records written in the master catalog. If WRITECHECK is specified, a record is written and then read, without data transfer, to test for the data check condition. A DASD cache will be bypassed. If NOWRITECHECK is specified, a record is written but no checking occurs.

Abbreviations: WCK and NWCK

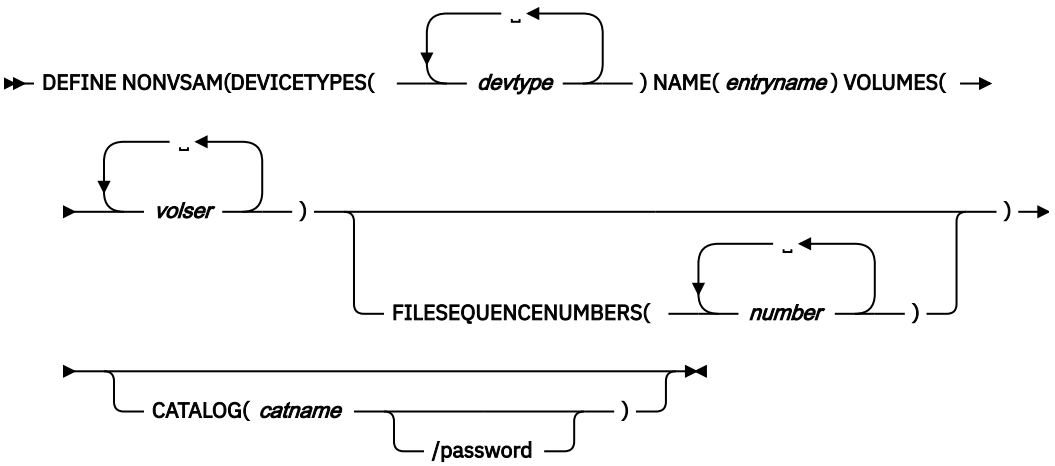
Default: NOWRITECHECK

DEFINE NONVSAM

The DEFINE NONVSAM command (DEF NVSAM) is used to catalog a nonVSAM file in a VSE/VSAM catalog. Any already existing file can be introduced into a master or user catalog through the DEFINE command. The only result of a DEFINE NONVSAM command is that an entry is created in a master or user catalog; no space is allocated or reserved. See “Example 4: Define NonVSAM and VSE/VSAM Files” on page 215.

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under “PARM” on page 205.)

The format of the command is:



DEFINE NONVSAM Parameters

CATALOG(catname)
identifies the catalog in which the nonVSAM file is to be defined. You do not have to specify this parameter (unless it is needed for password specification) when the nonVSAM file is to be defined in the default catalog. (The default catalog is (1) the job catalog if one is used, or (2) the master catalog.)

catname - is the name of the catalog.

password - specifies the update or higher-level password of the catalog if it is password-protected.

Abbreviation: CAT

DEVICETYPES(devtype)
specifies the device type(s) of the volumes containing the nonVSAM file. If the nonVSAM file resides on different device types, the device types must be specified in the same order as the volume serial numbers listed in the VOLUMES parameter.

Abbreviation: DEVT

DEVTYPE identifies any supported tape or disk device (for a list of supported devices, refer to the z/VSE System Control Statements).

- For tape devices specify 2400.
- For CKD and ECKD disk devices specify the IBM device number such as 3380 or 3390.
- NonVSAM files on FBA devices cannot be defined to a VSAM catalog.

FILESEQUENCENUMBERS(number)
specifies the file sequence number of the nonVSAM file being defined if it resides on tape. This number indicates the position of the file with respect to other files of the tape. If the file spans volumes, the file sequence number on every volume must be specified in the same order as the volumes in the VOLUMES parameter. This value can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

Abbreviation: FSEQN

NAME(entryname)
specifies the name of the nonVSAM file. The entryname is the name that appears in the catalog; it is the name used in all future references to the file. The entryname must be unique within the catalog in which it is defined. The name may consist of 1 through 44 alphameric characters, national characters (@, #, and $), and special characters (hyphen and 12-0 overpunch). Names containing more than eight characters must be segmented by periods; one through eight characters may be specified between periods. The first character of any name or name segment must be either an alphabetic or a national character. The last character in the name cannot be a period.

NONVSAM(options)
specifies that a nonVSAM file is to be defined. NONVSAM is followed by the parameters that describe the nonVSAM file; they are enclosed in parentheses.

Abbreviation: NVSAM

VOLUMES(volser)
specifies the volume(s) that are to contain the nonVSAM file. If the file resides on magnetic tape and there is more than one tape file belonging to the file on a single tape volume, you must repeat the volume's serial number in order to maintain a one-to-one correspondence between the volume serial numbers and the file sequence numbers in the FILESEQUENCENUMBERS parameter.

volser - may contain one through six alphameric, national (@, #, and $), and special characters (commas, blanks, semicolons, parentheses, slashes, asterisks, periods, quotation marks, ampersands, plus signs, hyphens, and equal signs). Single quotation marks within a volume serial number must be coded as two single quotation marks. For consistency with z/VSE job control, only alphameric characters should be used. A volume serial number must be enclosed in single quotation marks if it contains a special character.

Abbreviation: VOL

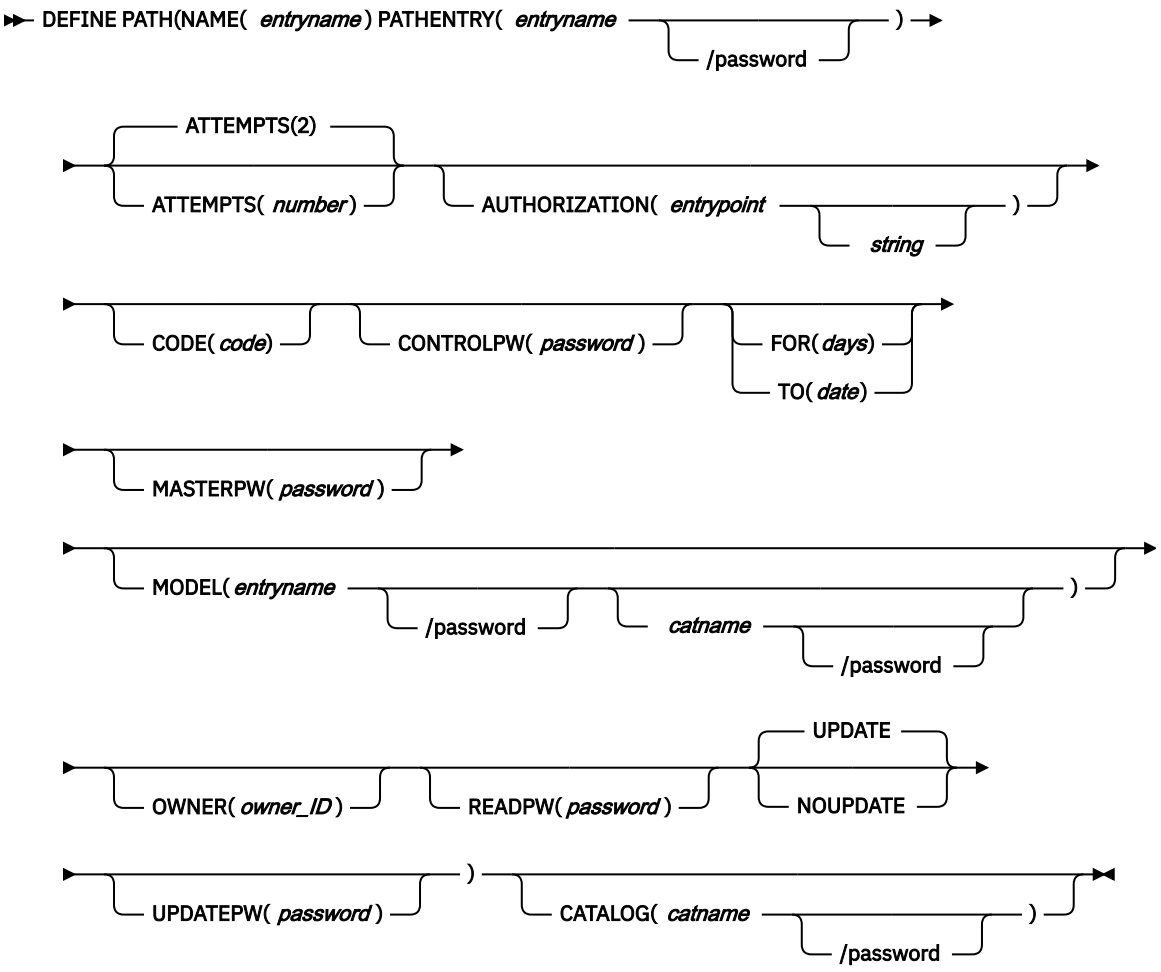
DEFINE PATH

The DEFINE PATH command (DEF PATH) is used to define a path. No data space is occupied by a path. A path relates an alternate index and its base cluster or, alternatively, a path may be defined directly over a base cluster. A path and its related alternate index and base cluster must all be defined in the same catalog. If the alternate index or base cluster specified in the PATHENTRY parameter is password-protected, the path being defined should also be password-protected to ensure the desired level of data security. Omitting password protection for the path allows access to the password-protected data without the need of specifying a password. For examples and more information, see:

- “Defining a Path” on page 35
- “Example 8: Creating an Alternate Index and Its Path” on page 223

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under “PARM” on page 205.)

The format of the DEFINE PATH command is:



DEFINE PATH Parameters: Summary

The parameters of the DEF PATH command are of the following categories:

Name, which specifies:

- The name of the path (NAME). This is a required parameter.

- The alternate index or base cluster considered as the entry to the path (PATHENTRY). This is a required parameter.
- The catalog that contains the alternate index or base cluster named in PATHENTRY (CATALOG).
- The path to be used as a model (MODEL). Refer to "Using an Object as a Model" in the VSE/VSAM User's Guide and Application Programming.

Allocation, which specifies:

- Whether the cluster's upgrade set is to be opened when the path (and its cluster) is opened (UPDATE, NOUPDATE).

Protection and integrity, which specifies:

- The passwords to be associated with the path (MASTERPW, CONTROLPW, UPDATEPW, READPW).
- A prompting code (CODE) and the number of attempts allowed to provide the correct password in response to prompting at the operator's console (ATTEMPTS).
- A user-supplied authorization verification routine (AUTHORIZATION).
- The owner of the path (OWNER).
- A retention period (FOR, TO).

DEFINE PATH Parameters

ATTEMPTS(number)
specifies the maximum number of times (0 through 7) the operator may try to enter a correct password in response to a prompting message. If 0 is specified, the operator is not prompted and is not allowed to enter a password from the console. This value can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

Abbreviation: ATT

Default: 2

AUTHORIZATION(entrypoint)
specifies that, in addition to passwords, you are supplying a USVR (user security verification routine) to check the authority of a processing program to access the path. VSE/VSAM transfers control to the USVR only after the program trying to open the path gives a correct password other than the master password. (The USVR is always bypassed whenever a correct master password is specified.) For more information, refer to the "User Security-Verification Routine" in the VSE/VSAM User's Guide and Application Programming.

entrypoint - specifies the entrypoint name may contain one through eight alphameric, national (@, #, and $), or special (the hyphen and 12-0 overpunch) characters. The first character must be an alphabetic or a national character. Entrypoint is the phase name in the sublibrary.

string - specifies up to 255 characters that are to be passed to the USVR when it receives control to verify authorization. The string must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a string must be coded as two single quotation marks if the string is enclosed in single quotation marks. If the string is specified in hexadecimal, it must be coded (X‘string’).

Abbreviation: AUTH

CATALOG(catname)
identifies the catalog that defines the alternate index or the base cluster named in the PATHENTRY parameter.

catname - specifies the name of the catalog. You do not have to specify this parameter (unless it is needed for password specification) when the entry is to be defined in the default catalog. (The default catalog is (1) the job catalog if one is used, or (2) the master catalog.)

password - specifies the update or higher-level password if the catalog is password-protected. If no password is specified and the catalog is password protected, VSE/VSAM asks the operator for the correct password.

Abbreviation: CAT

CODE(code)
specifies a code name for the path, if it is password-protected. If an attempt is made to access the path without a password, the code name is used in a prompting message to the operator rather than the name of the path. The operator can then provide the correct password. The code may contain one through eight EBCDIC characters.

The code must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. If a single quotation mark appears within the code, it must be coded as two single quotation marks. Code may also be coded in hexadecimal (X'n') form.

If CODE is not specified and an attempt is made to access a password-protected path without supplying a password, the operator is prompted with the name of the path. The operator can then provide the correct password.

CONTROLPW(password)
specifies a control password for the path. The control password permits read and write operations using CI access and all operations permitted by the update and read passwords. For more information, refer to the "Passwords to Authorize Access" in the VSE/VSAM User's Guide and Application Programming.

password - is a one through eight EBCDIC character password. If the password contains commas, semicolons, blanks, parentheses, or slashes, the password must be enclosed in single quotation marks. Single quotation marks within a password must be coded as two single quotation marks if the password is enclosed in single quotation marks. The password may also be coded in hexadecimal (X'password') form.

Abbreviation: CTLPW

FOR(days) | TO(date)
specifies the retention period for the path being defined. If neither TO nor FOR is specified, the path can be deleted at any time. Deleting it does not affect the existence of the alternate index and the base cluster.

FOR(days)
specifies the number of days for which the path is to be kept. If the number specified is 0 through 1830, the path is retained for the number of days specified; if the number is 1831 through 9999 (maximum), the expiration period of the path is considered as a never-expire date. The number of days can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

TO(date)
specifies the date until which the path is to be kept. The date has the form yyyyddd, where yyyy (GE current date through 2099) is the year and ddd (001 through 366) is the day. The new expiration date cannot be a past date and cannot be more than 99 years to the future. The old date form yyddd is also allowed for compatibility purposes. The yy value will be implicitly converted into a yyyy value, so that the date relates to the future. A value of the *yyddd* portion greater than 99365 (such as 99999, 1999999, or 2099999) will be considered to mean "retain indefinitely, does not expire".

MASTERPW(password)
specifies a master password for the path. If MASTERPW is not specified and if other passwords exist, the highest-level existing password automatically becomes the master password. The master password allows all operations. See CONTROLPW for the definition of password.

Abbreviation: MRPW

MODEL(entryname)
specifies that the entry of an already-defined path is to be used as a model for the entry being built. For more information about MODEL, refer to the "Using an Object as a Model" in the VSE/VSAM User's Guide and Application Programming.

entryname - specifies the name of the path entry to be used as a model.

password - specifies a password. If the path whose entry is to be used as a model is password-protected and is cataloged in a password-protected catalog, either the path's password or its catalog's password is required. If both the entry password and the catalog password are specified, the catalog password is used. If the protection attributes are to be copied, substitute the master password of either the path (following entryname) or its catalog (following catname). If the model's passwords are not to be copied, any password of either the path or its catalog can be used.

catname - specifies the name of the catalog in which the path whose entry is to be used as a model is defined. This parameter is required if (1) you are going to specify the password of the catalog that defines the path instead of specifying the password of the path itself, or (2) the catalog is not the default catalog. The default catalog is (1) the job catalog if one is used, or (2) the master catalog.

NAME(entryname)
specifies the name (file ID) of the path. The name may contain from 1 through 44 alphameric characters, national characters (@, #, and $), and special characters (the hyphen and the 12-0 overpunch). Names containing more than eight characters must be segmented by periods; one through eight characters may be specified between periods. The first character of any name or name segment must be either an alphabetic or a national character. The last character in the name cannot be a period.

NOUPDATE (see UPDATE)

OWNER(owner-id)
identifies the owner of the path. The owner ID may contain one through eight EBCDIC characters. The owner ID must be enclosed in single quotation marks if it contains a comma, semicolon, blank, parenthesis, or slash. If a single quotation mark appears within the owner ID, it must be coded as two single quotation marks when the owner ID is enclosed in single quotation marks. It may also be coded in hexadecimal form.

PATH(options)
specifies that a path must be defined. PATH is followed by other parameters.

PATHENTRY(entryname)
specifies the name (file ID) of either (a) the alternate index or (b) the base cluster that is to be considered as the entry to the path.

a. If the entry to the path is an alternate index, the termination of the path is the base cluster to which the alternate index is related (using the RELATE parameter of the DEFINE ALTERNATEINDEX command).
b. If the entry to the path is a base cluster, the name of the path is just another name for the base cluster, used to avoid unnecessary allocation of the upgrade set (see the UPDATE | NOUPDATE parameter). A path over a base cluster is called an alias. An alias lets you give a file an additional, equally valid name.

If the alternate index or the base cluster specified as entryname is protected, its master password is required. Alternatively, you can specify, in the CATALOG parameter, the master password of the catalog in which the alternate index or the base cluster is defined.

Abbreviation: PENT

READPW(password)
specifies a read password for the path. The read password permits read operations against the data records in the base cluster. See CONTROLPW for the definition of password.

Abbreviation: RDPW

TO (see FOR)

UPDATE | NOUPDATE
specifies whether the upgrade set belonging to the base cluster of the path is to be updated. If you specify NOUPDATE and the base cluster has one or more alternate indexes in its upgrade set, these alternate indexes will not be updated.

When a NOUPDATE path is opened for processing and the path entry is a base cluster, no alternate index needs to be available. No devices need to be allocated for any alternate index and no control blocks are built for them at OPEN time. When a NOUPDATE path is opened for processing and the path entry is an alternate index, only that alternate index needs to be available. No devices need to be allocated for other alternate indexes and no control blocks are built for them at OPEN time. Thus, NOUPDATE prevents unwanted allocation of the upgrade set even if the UPGRADE attribute is set for one or more of the base cluster's alternate indexes.

If you specify UPDATE, the upgrade set will be opened with the base cluster and be updated whenever the base cluster is modified, and devices have to be allocated for it.

Abbreviations: UPD and NUPD

Default: UPDATE

UPDATEPW(password)
specifies an update password for the path. The update password permits read and write operations against the data records in the base cluster. See CONTROLPW for the definition of password.

Abbreviation: UPDPW

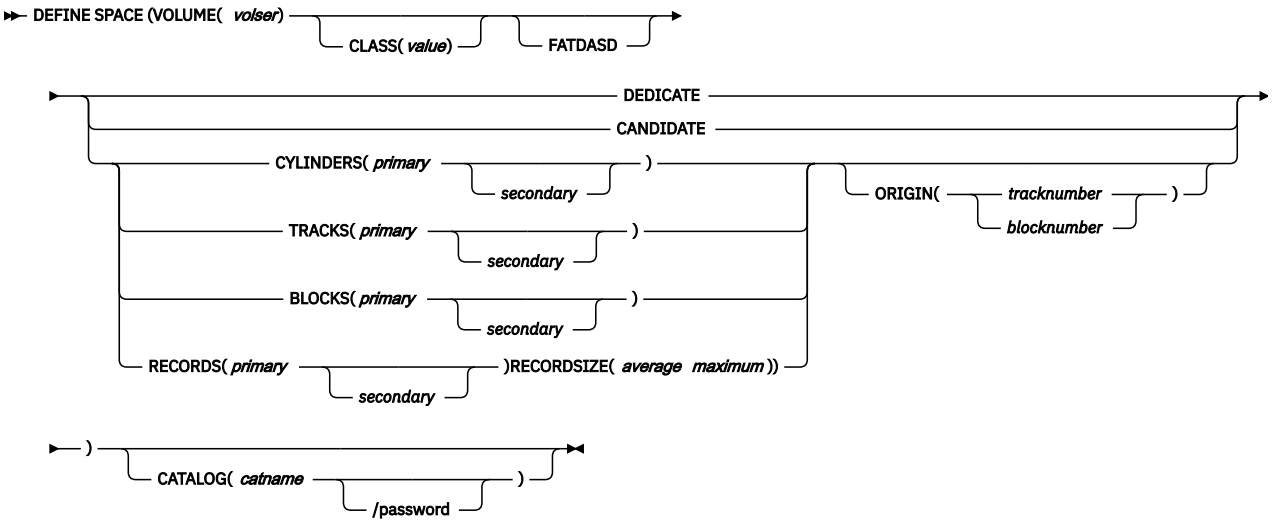
DEFINE SPACE

The DEFINE SPACE command (DEF SPC) is used to define VSE/VSAM data spaces, or to reserve a volume for future use by VSE/VSAM. A VSE/VSAM data space is space on a direct access volume that is owned and managed by VSE/VSAM. For examples and more information, see:

- "Defining a VSE/VSAM Data Space" on page 19
- "Example 2: Define a VSE/VSAM User Catalog and a VSE/VSAM Data Space" on page 210

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under "PARM" on page 205.)

The format of the DEFINE SPACE command is:



DEFINE SPACE Parameters

BLOCKS(primary [secondary])

specifies, for FBA devices only, the number of blocks to be allocated as data space. Each block is a fixed size of 512 bytes.

primary - specifies the number of blocks to be made available for primary and secondary allocation. Primary and secondary can be specified in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form. VSE/VSAM ignores the secondary value; however, it can be specified for z/OS compatibility.

Restriction: The maximum value of the BLOCKS parameter is 16,777,215 (X'FFFFFF'). If the specified value, rounded up to a minimum- or maximum-CA multiple, is larger than 16,777,215, an error message is issued and processing is stopped.

Note the following:

- If you specify a primary or secondary value that is not a multiple of a minimum CA, VSE/VSAM rounds it to a minimum CA multiple. For example, if you specify BLOCKS(40 20) for generic FBA, VSE/VSAM rounds 40 to 64 and 20 to 64, because minimum CA = 64. For SCSI devices, the minimum CA is 512 blocks. If you specify BLOCKS(40 20) for a SCSI device, VSE/VSAM rounds 40 to 512 and 20 to 512.

Data space can be defined by the ORIGIN and BLOCKS parameters. If the beginning block number does not coincide with a minimum CA multiple, VSE/VSAM rounds it up to the next minimum CA multiple; if the ending block (ORIGIN value plus BLOCKS value) does not coincide with a minimum CA multiple, VSE/VSAM rounds it down to the previous minimum CA multiple.

For example, ORIGIN (40) and BLOCKS (100) for generic FBA indicate that the data space begins at block number 64 (40 is rounded up to 64) and ends at block number 127 (40 + 100 - 1 = 139 is rounded down to 127.) Be aware that in some instances, after VSE/VSAM rounds the values you specified, zero space is the resultant allocation. For example, if you specify ORIGIN (40) and BLOCKS (50), VSE/VSAM rounds 40 up to 64 as before. However, when VSE/VSAM rounds the ending block value of 63 (40 + 24 -1 = 63) down to the closest minimum CA boundary, the value (63) is less than the beginning block value. Therefore, no data space is available for allocation (beginning block is 64, ending block is 63).

TRACKS or CYLINDERS cannot be specified for FBA devices; the RECORDS or DEDICATE parameters are valid for FBA devices. For further information on minimum CAs and FBA devices, refer to the "Minimum and Maximum CA Sizes" in the VSE/VSAM User's Guide and Application Programming.

The minimum primary space allocation for the initial data space on a volume is one maximum CA (cylinder).

secondary - the secondary value is not used in VSE/VSAM, but it may be specified for compatibility of the functional commands with z/OS.

Abbreviations: BLK or BLOCK

CANDIDATE

specifies that the volume named in the VOLUME parameter is reserved (is a candidate) for future use by VSE/VSAM. No space is allocated on the volumes. The volume will be available either to the catalog specified in the CATALOG parameter or, alternatively to the job or master catalog. Either the CANDIDATE parameter or the BLOCKS, TRACKS, CYLINDERS, DEDICATE, or RECORDS parameter must be specified.

CANDIDATE cannot be specified if:

- A VSE/VSAM catalog or VSE/VSAM data space already exists on the volume named in the VOLUMES parameter, or
- You specified the ORIGIN parameter.

Abbreviation: CAN

CATALOG(catname)
identifies the catalog in which the data space(s) are to be defined, or in which a candidate volume is to be indicated. You do not have to specify this parameter (unless it is needed for password specification) when the space is to be owned by the default catalog. (The default catalog is (1) the job catalog if one is used, or (2) the master catalog.)

Different catalogs can own space on one volume.

catname - specifies the name of the catalog.

password - specifies the update or higher-level password if the catalog is password-protected.

Abbreviation: CAT

CLASS(value)
Only one space classification can be specified for the total data space. You cannot specify CLASS if CANDIDATE is specified. The values you can specify are:

0
Data spaces defined under DFSMSdfp VSAM are treated as class 0 which is also the default.

1-7
Specifies that the defined data space is classified according to your own criteria. You can use any criteria to classify the data space; for example, you can assign particular files to the middle section of a volume if you feel that performance will be enhanced.

For more information on space classes, refer to the "Data Space Classification" in the VSE/VSAM User's Guide and Application Programming.

CYLINDERS(primary [secondary])
specifies the number of cylinders to be allocated as data space. The value can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B“n”) form. Either this parameter or the CANDIDATE, BLOCKS, RECORDS, DEDICATE, or TRACKS parameter must be specified. Do not specify CYLINDERS or TRACKS for FBA devices or BLOCKS for CKD devices.

primary - specifies the number of cylinders available for primary allocation. You can specify the ORIGIN parameter to indicate the beginning point of your data space.

secondary - the secondary value is not used in VSE/VSAM, but it may be specified for the compatibility of the functional command with z/OS.

It is recommended to use this parameter to allocate space for Large DASD.

Abbreviation: CYL

DEDICATE
specifies that the unowned and unallocated data space (maximum of 16 extents) on VOLUMES (volser) is to be allocated to VSE/VSAM. DEDICATE is mutually exclusive with the space allocation parameters (BLOCKS, CANDIDATE, RECORDS, TRACKS, CYLINDERS). Do not specify the ORIGIN parameter with DEDICATE.

A maximum of 16 extents can be acquired by VSE/VSAM for every individual volume; a maximum of 255 extents can be acquired by VSE/VSAM for a given DEFINE SPACE command.

Abbreviation: DED

FATDASD
FATDASD enables VSAM to allocate up to 65520 cylinders on the specified volume. This definition cannot be changed until the associated catalog is deleted and redefined. The VSAM space bitmap is managed in cylinders instead of tracks.

Specified volumes which are unknown to the current VSAM catalog but have a minimum real capacity of 64K tracks, are defined as BIG-DASDs (max 10017 cylinders) if parameter FATDASD has been omitted.

Abbreviations: FD and FAT

ORIGIN(tracknumber | blocknumber)
specifies the beginning point (track number or block number) of the data space. (If the block number does not coincide with a minimum CA boundary, VSE/VSAM rounds it up to the next minimum CA boundary.) See the BLOCKS parameter for a description of rounding. VSE/VSAM determines the ending point of the data space by adding the value you specify for CYLINDERS, BLOCKS, TRACKS, or RECORDS (VSE/VSAM converts records and blocks to minimum CAs) to the ORIGIN specification.

Specify a value greater than 0 for tracknumber and a value greater than 1 for blocknumber.

If you specify ORIGIN, do not specify the CANDIDATE parameter. You can omit both the DEDICATE and ORIGIN parameters; in this case the beginning point of the data space is the first available area on the volume that is large enough to contain your primary allocation (if you specified CYLINDERS, the beginning boundary is a cylinder boundary).

Abbreviation: ORG

RECORDS(primary [secondary])
specifies the number of records for which space is to be allocated as data space. The RECORDSIZE parameter gives the size used (average record size) for calculating the amount of space.

Restriction:

In the RECORDS parameter, you can specify a maximum value of 16,777,215 (X'FFFFFF').

See the CYLINDERS parameter for the information and restrictions that apply to CKD devices. See the BLOCKS parameter for information on FBA devices.

RECORDSIZE must be specified if this parameter is specified.

Abbreviation: REC

RECORDSIZE(average maximum)
specifies the average and maximum record size, in bytes, to be used to calculate the amount of space to be allocated in terms of number of records. These values can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

RECORDSIZE must be specified if RECORDS is specified. The average size that can be specified is 1; the maximum size is 32,761 bytes.

Abbreviation: RECSZ

SPACE(options)
specifies that a data space is to be defined.

Abbreviation: SPC

TRACKS(primary [secondary])
specifies the number of tracks to be allocated as data space. See the CYLINDERS parameter for more information and restrictions.

For Large DASD, allocation in CYLINDERS is recommended.

Abbreviation: TRK

VOLUME(volser)
specifies the volume on which data space is to be defined or which is to be candidate available to the catalog indicated in the CATALOG parameter. You cannot specify more than one volume.

On the indicated volume, either the amount specified in the space allocation parameter (BLOCKS, etc.) is allocated, or, if DEDICATE is specified, all unowned and unallocated data space (maximum of 16 extents) is allocated to VSE/VSAM. If CANDIDATE is specified, the volume is merely marked as being available to the catalog. If a volume is larger than 65535 tracks (min-CAs, respectively), VSE/VSAM can place data only within the first 65535 tracks (or min-CAs), rounded down to the next cylinder (max-CA) boundary.

A volume serial number, volser, may contain one through six alphameric, national (@, #, and $), and special characters (commas, blanks, semicolons, parentheses, slashes, asterisks, periods, quotation marks, ampersands, plus signs, hyphens, and equal signs). Single quotation marks within a volume

serial number must be coded as two single quotation marks. A volume serial number must be enclosed in single quotation marks if it contains a special character. For consistency with z/VSE job control, only alphameric characters should be used.

The volume to be used can be on *virtual disk*, but note that the catalog and the object to be defined must both reside on virtual disk.

Abbreviation: VOL

DEFINE USERCATALOG

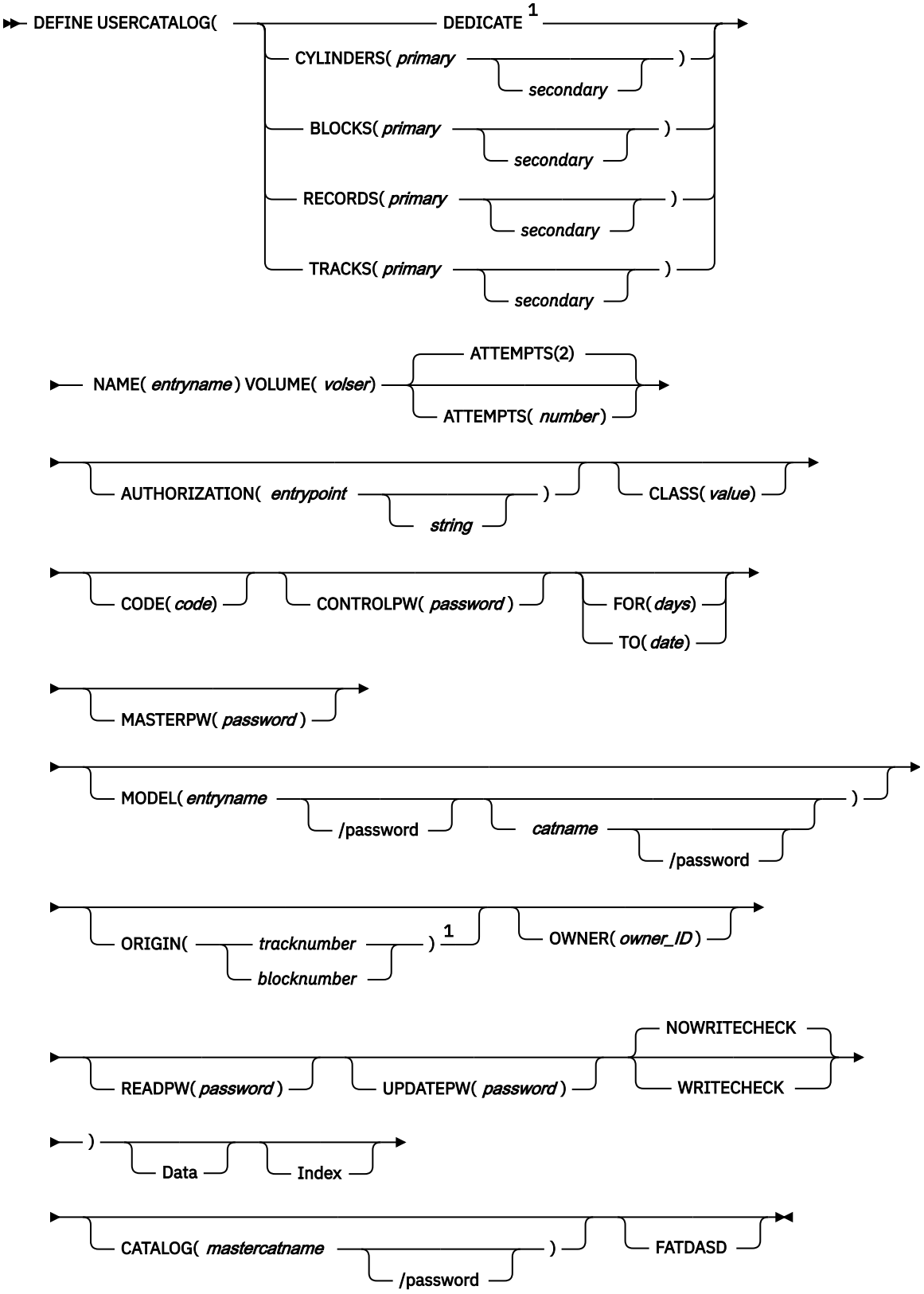
The DEFINE USERCATALOG command (DEF UCAT) is used to define a user catalog. When you define a user catalog, a data space to contain the catalog is automatically created. An entry for the volume containing the data space is placed in the user catalog. A catalog must have a master password if any VSE/VSAM files cataloged in it are to be password-protected. A catalog must have an update or higher-level password if any nonVSAM files cataloged in it are to be password-protected. For examples and more information, see:

- “Defining Objects in a Catalog” on page 15
- “Example 1: Define a System's Catalogs” on page 209
- “Example 2: Define a VSE/VSAM User Catalog and a VSE/VSAM Data Space” on page 210

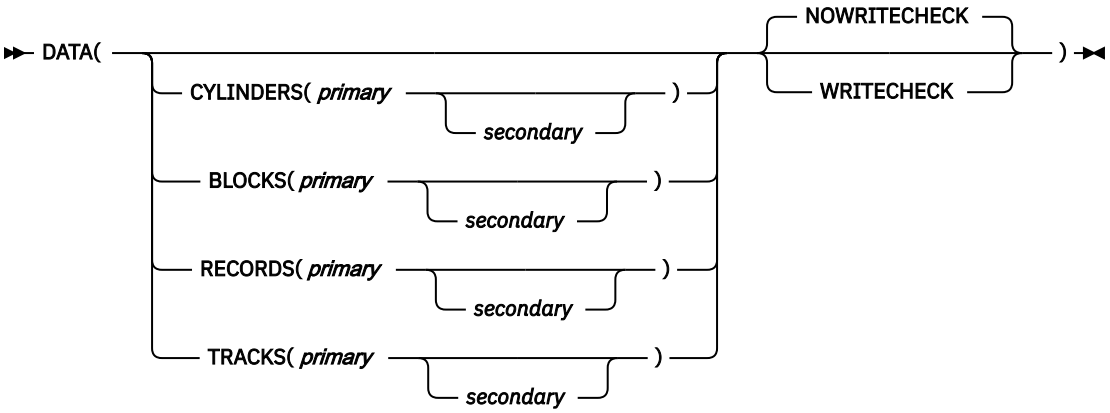
If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under “PARM” on page 205.)

Whenever a user catalog is defined on a virtual disk and the virtual disk is lost for VSE/VSAM (for example, on re-IPL or detach), you have to EXPORT DISCONNECT the corresponding user catalog before defining a new user catalog of the same volume serial name.

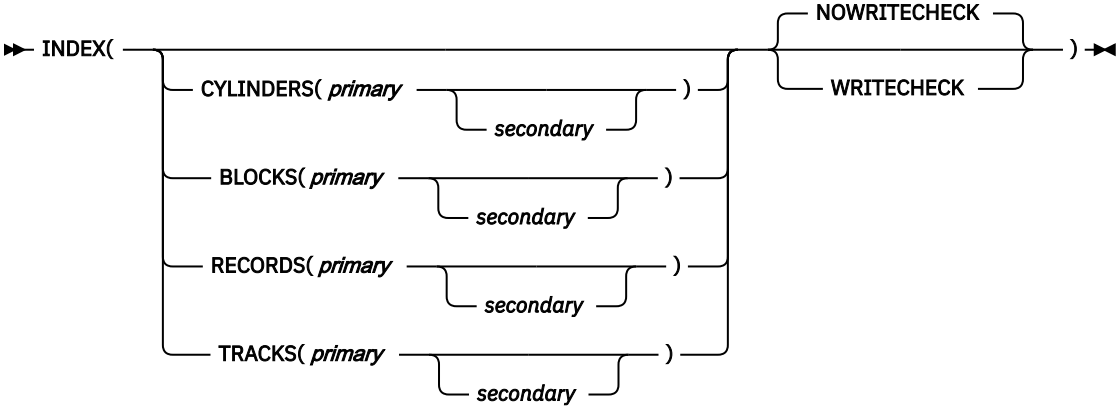
The format of the command is shown below. Note that the organization of the command consists of three groupings. These groupings are referred to as levels in this publication. Note that some parameters appear in more than one level. (The optional CATALOG parameter stands alone; it does not belong to any level.)


DEFINE USERCATALOG(
DEDICATE
1
CYLINDERS(primary)
secondary
BLOCKS(primary)
secondary
RECORDS(primary)
secondary
TRACKS(primary)
secondary
NAME(entryname) VOLUME(volser)
ATTEMPTS(2)
ATTEMPTS(number)
AUTHORIZATION(entrypoint)
string
CLASS(value)
CODE(code)
CONTROLPW(password)
FOR(days)
TO(date)
MASTERPW(password)
MODEL(entryname)
/password
catname
/password
ORIGIN(tracknumber)
blocknumber
1
OWNER(owner_ID)
NOWRITECHECK
READPW(password)
UPDATEPW(password)
WRITECHECK
)
Data
Index
CATALOG(mastercatname)
/password
FATDASD

Data



Index



Notes:

[1] DEDICATE and ORIGIN are mutually exclusive.

DEFINE USERCATALOG Parameters: Summary

The parameters of the DEF UCAT command can be divided into the following categories:

Name, which specifies:

- The name of the catalog being defined (NAME). This is a required parameter.
- The name of the master catalog that points to the user catalog (CATALOG).
- The name of the master or user catalog that is to be used as a model for this user catalog (MODEL). Refer to the "Using an Object as a Model" in the VSE/VSAM User's Guide and Application Programming.

Allocation, which specifies:

- The volume and location on the volume where the catalog is to reside (ORIGIN, VOLUME).
- The amount of space to be allocated (BLOCKS, CYLINDERS, DEDICATE, RECORDS, TRACKS). One of these parameters is required.
- The classification of the catalog's data space (CLASS). Refer to the "Data Space Classification" in the VSE/VSAM User's Guide and Application Programming.
- The FATDASD parameter, it has to be explicitly specified, otherwise VSAM handles a DASD with more than 64K tracks as BIG-DASD (limited to 10017 cylinders).

Protection and integrity, which specifies:

- Passwords to be associated with the catalog (MASTERPW, CONTROLPW, UPDATEPW, READPW).
- A prompting code (CODE) and number of attempts allowed to provide the correct password in response to prompting at the operator's console (ATTEMPTS).

- A user-supplied security verification routine (AUTHORIZATION).
- The owner of the catalog (OWNER).
- A retention period (FOR, TO).
- Whether write-check operations are to be performed as records are inserted in the catalog (WRITECHECK, NOWRITECHECK).

DEFINE USERCATALOG Parameters

ATTEMPTS(number)
specifies the maximum number of times (0 through 7) the operator can try to enter a correct password in response to a prompting message. If 0 is specified, the operator is not prompted and is not allowed to enter a password from the console. This value can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

Abbreviation: ATT

Default: 2

AUTHORIZATION(entrypoint[string])
specifies that, in addition to passwords, you are supplying a routine to check the authority of a processing program to access the user catalog. VSE/VSAM transfers control to the verification routine only after the program trying to open the catalog gives a correct password other than the catalog's master password. (The verification routine is always bypassed whenever a correct master password is specified.) Refer to the "Data Protection" in the VSE/VSAM User's Guide and Application Programming.

entrypoint - specifies the entry point name of the user's security verification routine. The entry point name may contain one through eight alphameric, national (@, #, and $), or special (the hyphen and 12-0 overpunch) characters. The first character must be an alphabetic or a national character.

string - specifies up to 255 characters that are to be passed to the user's security verification routine when it receives control to verify authorization. The string must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a string must be coded as two single quotation marks if the string is enclosed in single quotation marks. If string is specified in hexadecimal, it must be preceded by X and enclosed in single quotation marks (X‘string’).

Abbreviation: AUTH

BLOCKS(primary [secondary])
specifies, for FBA devices only, the number of blocks to be suballocated to the user catalog. Each block is a fixed size of 512 bytes. This parameter must be specified at the catalog level. In addition, it can be specified at the data component level, or at both the data and index component levels.

primary - specifies the number of blocks to be made available for primary and secondary allocation. Primary and secondary can be specified in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

Restriction: The maximum value of the BLOCKS parameter is 8,388,096 (X'7FFFFF'). If the specified value, rounded up to a minimum- or maximum-CA multiple, is larger than 8,388,096, an error message is issued and processing is stopped.

The data component requires space for at least three CAs.

Note the following:

- If you specify a primary or secondary value that is not a multiple of a minimum CA, VSE/VSAM rounds it to a minimum CA multiple. For example, if you specify BLOCKS(40 20) for generic FBA, VSE/VSAM rounds 40 to 64 and 20 to 64, because minimum CA = 64.

A further consideration applies to the BLOCKS specification at the catalog level only. VSE/VSAM acquires the data space for the catalog concurrently with the DEFINE USERCATALOG command. If the beginning block number (specified in the ORIGIN parameter) does not coincide with a minimum CA boundary, VSE/VSAM rounds it up to the next minimum CA boundary; if the ending block (ORIGIN

value plus BLOCKS value) does not coincide with a minimum CA boundary, VSE/VSAM rounds it down to the previous minimum CA boundary. Again, if you request more than a maximum CA, VSE/VSAM rounds to a maximum CA boundary.

For example, ORIGIN (40) and BLOCKS (100) for generic FBA indicates that space for the catalog begins at block number 64 (40 is rounded to 64) and ends at block number 127 (40 + 100 - 1 = 139, rounded down to 127).

Be aware that, in some instances, after VSE/VSAM rounds the values you specified, zero space is the resultant allocation. For example, if you specify ORIGIN (40) and BLOCKS (50), VSE/VSAM rounds 40 up to 64 as before. However when VSE/VSAM rounds the ending block value of 89 (40 + 50 -1 = 89) down to the closest minimum CA boundary, the value (63) is less than the beginning block value. Therefore no data space is available for allocation to the catalog (the beginning block is 64 and the ending block is 63).

Secondary allocation of space to the catalog depends on the following:

- If you do not specify a secondary value, VSE/VSAM itself extends the catalog's data and index components an additional 15 times (one CA at a time) if needed.
- If you specify secondary at the data component level, VSE/VSAM uses the specified value to extend (if needed) your catalog an additional 15 times. Note that if the value you specify does not coincide with a CA boundary, VSE/VSAM rounds it down to the nearest multiple (CA) and then uses this multiple for the secondary extensions.
- If you specify a secondary value at the catalog level or index component level, VSE/VSAM ignores it (in this case the specified value is solely an entry in the catalog for purposes of z/OS compatibility) and instead extends the index component by one CA at a time for an additional 15 times if needed.

For SCSI devices, explicit space allocation is required if the ORIGIN parameter is used. The minimum primary allocation is 3072 blocks. For FBA devices, it is recommended not to use the VSAM default settings but rather to explicitly define VSAM catalogs with larger primary and secondary allocation sizes. The default catalog on an FBA device will have 256 blocks as primary and 128 blocks as secondary allocation. These values will result in a catalog-full condition after defining 512 files (including the catalog itself). The catalog has already reached 16 extents (the maximum for VSAM catalogs) and cannot be extended further.

TRACKS or CYLINDERS cannot be specified for FBA devices; the RECORDS or DEDICATE parameters are valid for FBA devices. For further information on minimum CAs and FBA devices, refer to the "Minimum and Maximum CA Sizes" in the VSE/VSAM User's Guide and Application Programming. For more information on space allocation, see “How Data Space is Assigned to a Catalog” on page 16.

Abbreviations: BLK or BLOCK

CATALOG(mastercatname)
specifies the name and update or higher-level password of the master catalog, if the master catalog is password-protected. The password must be provided in this parameter or in response to prompting.

Abbreviation: CAT

CLASS(value)
specifies the data space classification to be assigned to the total amount of data space allocated for the catalog (even that portion of space not used, if any, by the catalog). The values you can specify (only at the catalog level) are:

0
Specifies the standard space class. Data spaces defined under DFSMSdfp VSAM are treated as class 0 which is also the default.

1-7
Specifies that the defined data space is classified according to your own criteria. You can use any criteria to classify the data space; for example, you can assign particular files to the middle section of a volume if you feel that performance will be enhanced.

For more information on space classes, refer to the "Data Space Classification" in the VSE/VSAM User's Guide and Application Programming.

CODE(code)
specifies a code name for the user catalog. If an attempt is made to access a password-protected catalog without a password, the code name is used in a prompting message to the operator rather than the name of the catalog. The operator can then provide the correct password.

The code may contain from one through eight EBCDIC characters. The code must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. If a single quotation mark appears within the code, it must be coded as two single quotation marks when the code is enclosed in single quotation marks. Code can also be expressed in hexadecimal (X‘code’) form.

If CODE is not specified and an attempt is made to access a password-protected catalog without supplying a password, the operator is prompted with the name of the user catalog.

CONTROLPW(password)
specifies a control password for the user catalog. The control password permits read and write operations using control interval access and all operations permitted by the update and read passwords. Because the master password is required to open the catalog as a file, the CONTROLPW has little meaning for a catalog. Refer to the "Passwords to Authorize Access" in the VSE/VSAM User's Guide and Application Programming.

password - is a one through eight EBCDIC character password. If the password contains commas, semicolons, blanks, parentheses, or slashes, the password must be enclosed in single quotation marks. Single quotation marks within a password must be coded as two single quotation marks if the password is enclosed in single quotation marks. The password can also be expressed in hexadecimal (X‘password’) form.

Note: Authorization checking for files cataloged in the user catalog is performed only if the catalog is password-protected.

Abbreviation: CTLPW

CYLINDERS(primary [secondary])
specifies, for CKD devices only, the number of cylinders to be allocated. Either this parameter or the BLOCKS, DEDICATE, RECORDS, or TRACKS parameter must be specified at the user catalog level. In addition, they (except DEDICATE) can be specified at the data level, or at both the data and index levels. Do not specify CYLINDERS or TRACKS for FBA devices, or BLOCKS for CKD devices. See “How Data Space is Assigned to a Catalog” on page 16 for more information. See the ORIGIN parameter for boundary requirements.

primary - specifies the number of cylinders available for primary and secondary allocation. These values can be expressed in decimal (n), hexadecimal (X‘n’), or binary (B‘n’) form.

The primary allocation for the combined data and index components must be contained in a single extent (contiguous data space). The data component requires space for at least three CAs.

Secondary suballocation of space to the catalog depends on the following:

- If you specify secondary at the data component level, VSE/VSAM uses the specified value to extend (if needed) your catalog an additional 15 times. Note that if the value you specify does not coincide with a CA boundary, VSE/VSAM rounds it down to the nearest multiple (CA) and then uses this multiple for the secondary extensions.
- If you specify a secondary value at the catalog level or index component level, VSE/VSAM ignores it (in this case the specified value is solely an entry in the catalog for purposes of z/OS compatibility) and instead extends the index component by one CA at a time for an additional 15 times if needed.
- If you do not specify a secondary value, VSE/VSAM itself extends the catalog's data and index components an additional 15 times (one CA at a time) if needed.

It is recommended to use this parameter for Large DASD; the minimum value is 5 cylinders for BIG- and for FAT-DASD. The maximum value for primary and secondary allocation is 5000 cylinders.

Abbreviation: CYL

DATA(options)
specifies attributes of the data component of the catalog. The allocation of space for the data and index components of the user catalog is discussed in "How Data Space is Assigned to a Catalog" on page 16. Buffer space and write-checking may also be specified for the data component.

Note: If DATA allocation parameters are not coded, the space specified at the user catalog level is available to contain catalog entries only. The total amount of space specified through DATA and INDEX parameters cannot be greater than that specified at the USERCATALOG level.

DEDICATE
specifies that up to 16 extents of the unowned and unallocated space on VOLUME(volser) is to be owned by the user catalog. You must delete expired files from the VTOC before VSE/VSAM will allocate that space on the VTOC. DEDICATE is mutually exclusive with the space allocation parameters (BLOCKS, RECORDS, TRACKS, CYLINDERS) and can be specified at the catalog level only. Do not specify the ORIGIN parameter with DEDICATE. There must be a contiguous area of space on the volume large enough to contain the primary allocation.

You can specify:

- DEDICATE alone - no space parameters (BLOCKS, TRACKS, RECORDS, or CYLINDERS) are specified at the data and index component levels; in this case the data space suballocated to the catalog itself is calculated by VSE/VSAM to be the minimum size catalog and the remaining space is available for later use by VSE/VSAM.
- DEDICATE at the catalog level and a space parameter value at both the data component and index component levels; in this case the data space suballocated to the catalog itself, is the sum of the values specified at the data and index component levels. Any VSE/VSAM data space that is not suballocated at this time is available for later use by VSE/VSAM.
- DEDICATE at the catalog level and a space parameter value at the data component level only; in this case VSE/VSAM calculates the space required for the index component and adds this amount to the data component specification. This sum becomes the amount of data space that is suballocated to the catalog. Any VSE/VSAM data space that is not suballocated at this time is available for later use by VSE/VSAM.

DEFINE USERCATALOG with option DEDICATE on a BIG- or FAT-DASD (ECKD >= 65535 tracks) will be rejected, if previously existing VTOC entries would cause more than one VSAM space extent to be generated.

Abbreviation: DED

FATDASD
FATDASD enables VSAM to allocate up to 65520 cylinders on the specified volume. This definition cannot be changed until this catalog is deleted and redefined. The VSAM space bitmap will be managed in cylinders instead of tracks.

Specified volumes which are unknown to the current VSAM catalog but have a minimum real capacity of 64K tracks, are defined as BIG-DASDs (max 10017 cylinders) if option FATDASD has been omitted.

Abbreviations: FD and FAT

FOR(days) | **TO(date)**
specifies the retention period for the user catalog.

FOR(days)
specifies the number of days the user catalog is to be kept. If the number specified is 0 through 1830, the catalog is retained for the number of days specified; if the number is 1831 through 9999 (maximum), the expiration date of the catalog is considered as a never-expire date. This value can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

TO(date)
specifies the date until which the user catalog is to be kept. The date has the form yyyyddd, where yyyy (GE current date through 2099) is the year and ddd (001 through 366) is the day. The new expiration date cannot be a past date and cannot be more than 99 years to the future. The old date form yyddd is also allowed for compatibility reasons. The yy value will be implicitly converted

into a yyyy value, so that the date relates to the future. A value of the *yyddd* portion greater than 99365 (such as 99999, 1999999, or 2099999) will be considered to mean "retain indefinitely, does not expire".

Default: The expiration data is set to 0, which means that the user catalog can be deleted whenever it is empty.

INDEX(options)
specifies attributes of the index component of the catalog. The allocation of space for the data and index components of the user catalog is discussed in "How Data Space is Assigned to a Catalog" on page 16.

An allocation parameter (BLOCKS, CYLINDERS, RECORDS, or TRACKS) must have been specified as a parameter of DATA for an allocation parameter to be specified as a parameter of INDEX. The total amount of space specified through DATA and INDEX subparameters cannot be greater than that specified at the USERCATALOG level.

The WRITECHECK/NOWRITECHECK parameter defaults to whatever was specified for the user catalog. If the allocation parameters are not specified, VSE/VSAM calculates the amount of space for the index component.

The primary index allocation will be at least 4% of the primary data allocation, and the secondary index allocation will be the same as the primary allocation.

Abbreviation: IX

MASTERPW(password)
specifies a master password for the user catalog. If MASTERPW is not specified and if other passwords exist, the highest level existing password automatically becomes the master password. The master password allows all operations; it is required to open the catalog as a file. See CONTROLPW for the definition of password.

Abbreviation: MRPW

MODEL(entryname)
specifies that an existing user or master catalog is to be used as a model for the user catalog being defined. When one entry is used as a model for another, its attributes are copied as the new entry is defined. You may use some attributes of the model and override others by explicitly specifying them in the definition of the user catalog.

If you specify the MODEL parameter you must specify the following parameters in the DEFINE command even if no attributes of the model are to be changed or added:

- Space allocation parameter (BLOCKS, CYLINDERS, DEDICATE, RECORDS, or TRACKS) at the user catalog level.
- NAME(entryname)
- ORIGIN(tracknumber | blocknumber) - specified if DEDICATE not specified or you do not want the "default ORIGIN".
- VOLUME(volser)

Note that the space suballocated to the catalog itself is controlled by:

- The space allocation attributes previously established by the model catalog, or, by
- The explicit specification of space parameters (in the current DEFINE USERCATALOG command) at the data or index component level (an explicit specification overrides the model attributes).

For more information about MODEL, refer to the "Using an Object as a Model" in the VSE/VSAM User's Guide and Application Programming.

entryname - specifies the name of the master or user catalog to be used as a model.

password - specifies a password of the catalog to be used as a model if it is password-protected. If both the entry password and the catalog password are specified, the catalog password is used. If

the protection attributes are to be copied, you must specify the master password. If the protection attributes are not to be copied, you may specify any password.

catname - specifies the name of the catalog to be used as a model. This parameter is required if (1) you are going to specify the password of the catalog that contains the entry instead of specifying the password of the entry itself, or (2) the catalog is not the default catalog. (The default catalog is (1) the job catalog if one is used, or (2) the master catalog.)

The VOLUME parameter cannot be modeled and must be specified to uniquely identify the volume.

Your job may be terminated because of allocation problems if you use the MODEL parameter and specify a device type different from that specified for the model through the VOLUME parameter.

NAME(entryname)
specifies the name of the catalog. The name may contain from 1 through 44 alphameric characters, national characters (@, #, and $), and special characters (the hyphen and the 12-0 overpunch). Names containing more than eight characters must be segmented by periods; one through eight characters may be specified between periods. The first character of any name or name segment must be either an alphabetic or national character. The last character in the name cannot be a period.

NOWRITECHECK (see WRITECHECK)

ORIGIN(tracknumber | blocknumber)
specifies the beginning point (track number or block number) of the user catalog's data space. (If the block number does not coincide with the minimum CA boundary, VSE/VSAM rounds it up to the next minimum CA boundary. See the BLOCKS parameter for a description of rounding.) VSE/VSAM determines the ending point of the data space by adding the value you specify (at the user catalog level) for CYLINDERS, BLOCKS, TRACKS, or RECORDS (VSE/VSAM converts records and blocks to minimum CAs) to the ORIGIN specification. For examples, see “Using the ORIGIN Parameter for a Catalog” on page 18.

Specify ORIGIN at the catalog level only. Specify a value greater than 0 for tracknumber and a value greater than 1 for blocknumber. You can omit the ORIGIN (and DEDICATE) parameter; in this case the beginning point of the catalog's data space is the first available area on the volume that is large enough to contain your primary allocation (if you specified CYLINDERS, the beginning boundary is a cylinder boundary).

If ORIGIN is specified for a SCSI device, the primary space allocation must be explicitly specified with at least 3072 blocks.

Abbreviation: ORG

OWNER(owner-id)
identifies the owner of the catalog. The owner ID may contain one through eight EBCDIC characters and be enclosed in single quotation marks if it contains a comma, semicolon, blank, parenthesis, or slash. If a single quotation mark appears within the owner ID it must be coded as two single quotation marks if the owner ID is enclosed in single quotation marks. It can also be coded in hexadecimal form.

READPW(password)
specifies a read password for the catalog. The read password permits read operations. See CONTROLPW for the definition of password.

Abbreviation: RDPW

RECORDS(primary [secondary])
specifies the number of records for which space is to be allocated. Every record in the catalog contains 512 bytes.

Restriction:
In the RECORDS parameter, you can specify a maximum value of 16,777,215 (X'FFFFFF').

See the CYLINDERS parameter for the information and restrictions that apply to CKD devices. See the BLOCKS parameter for information on FBA devices.

Abbreviation: REC

TO (see FOR)

TRACKS(primary [secondary])
specifies the number of tracks to be allocated. See the CYLINDERS parameter for more information and restrictions.

Abbreviation: TRK

UPDATEPW(password)
specifies an update password for the user catalog. The update password permits read and write operations. See CONTROLPW for the definition of password.

Abbreviation: UPDPW

USERCATALOG(options)
specifies that a user catalog is to be defined. USERCATALOG is followed by the parameters specified for the catalog as a whole; they are optionally followed by the DATA and/or INDEX parameters and their subparameters.

Abbreviation: UCAT

VOLUME(volser)
specifies the volume that is to contain the catalog. No other catalog may reside on this volume.

The volume serial number, volser, may contain one through six alphameric, national (@, #, and $), and special characters (commas, blanks, semicolons, parentheses, slashes, asterisks, periods, quotation marks, ampersands, plus signs, hyphens, and equal signs). Single quotation marks within the volume serial number must be coded as two single quotation marks. The volume serial number must be enclosed in single quotation marks if it contains a special character. For consistency with z/VSE job control, only alphameric characters should be used.

User catalogs may reside on a virtual disk.

Abbreviation: VOL

WRITECHECK | NOWRITECHECK
specifies whether to check the data transfer of records written in the user catalog. If WRITECHECK is specified, a record is written and then read, without data transfer, to test for the data check condition. If NOWRITECHECK is specified, a record is written but no checking occurs.

Abbreviations: WCK and NWCK

Default: NOWRITECHECK

DELETE

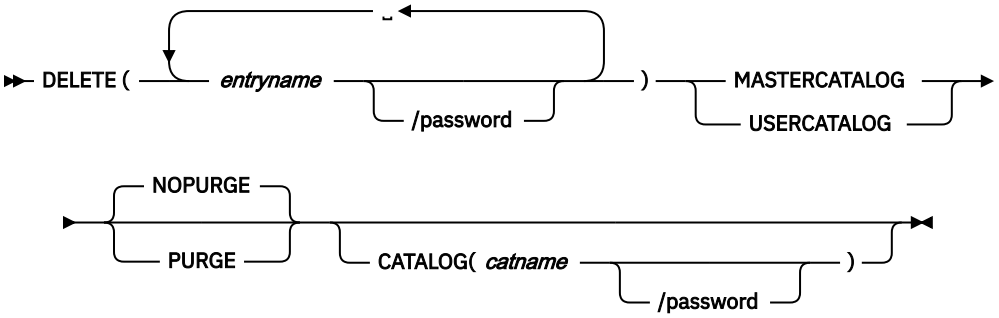
The DELETE command (DEL) is used to delete entries (for any previously-defined objects) from a catalog. This, in effect, causes the objects to cease to exist. When the entry represents an object that occupies space in a VSE/VSAM data space, the object's space is made available for other VSE/VSAM objects. For examples and more information, see

- “Deleting Catalog Entries” on page 39
- “Example 18: Deleting Entries in a User Catalog and the User Catalog Itself” on page 233
- “Example 24: Deleting Entries in the Master Catalog and the Master Catalog Itself” on page 237

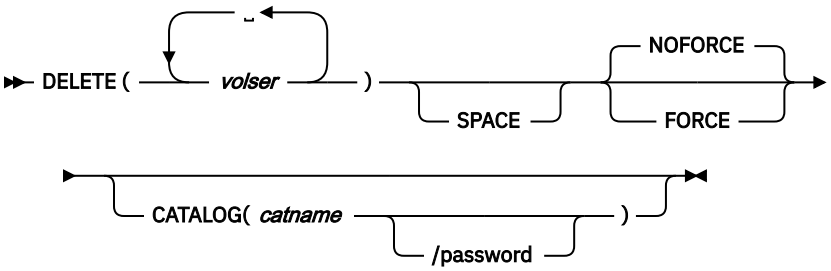
If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under “PARM” on page 205.)

DELETE can be used for several tasks, depending on what you are deleting: a catalog, a space, a cluster, an alternate index, a path, or a NonVSAM object. For each task different options are used. The following sections show the formats for the various tasks. The parameter descriptions have been summarized after the format descriptions.

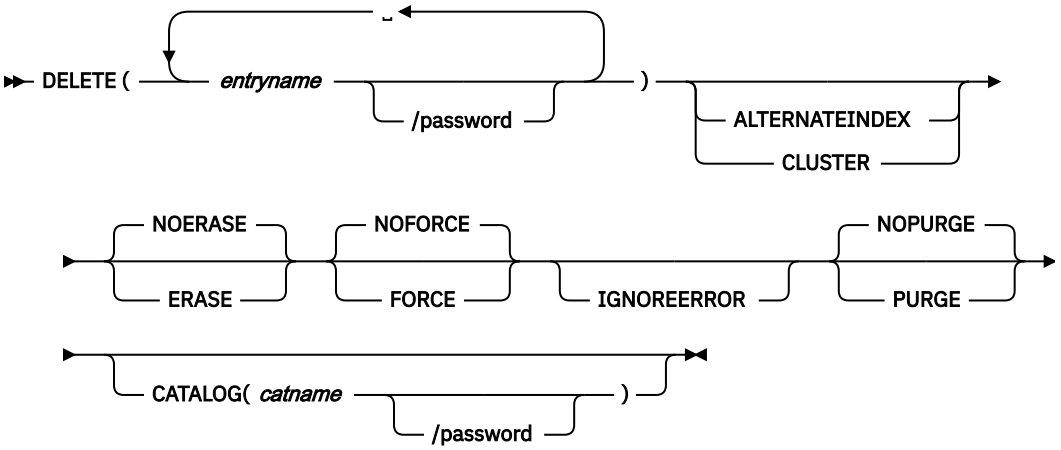
Format 1: Deleting a Catalog



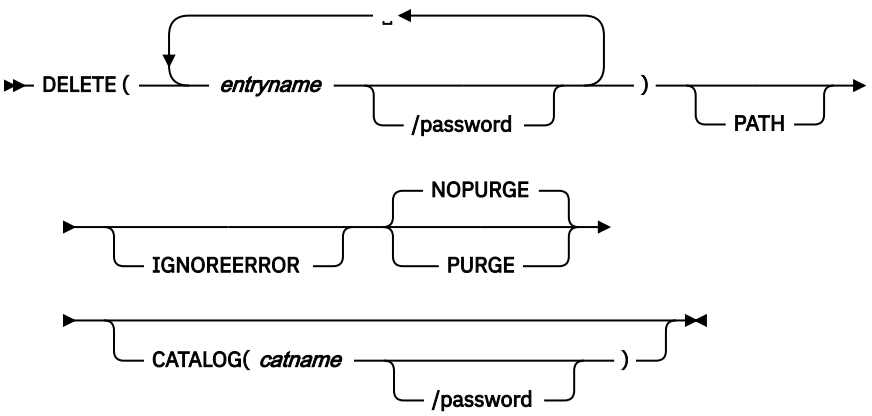
Format 2: Deleting a Space



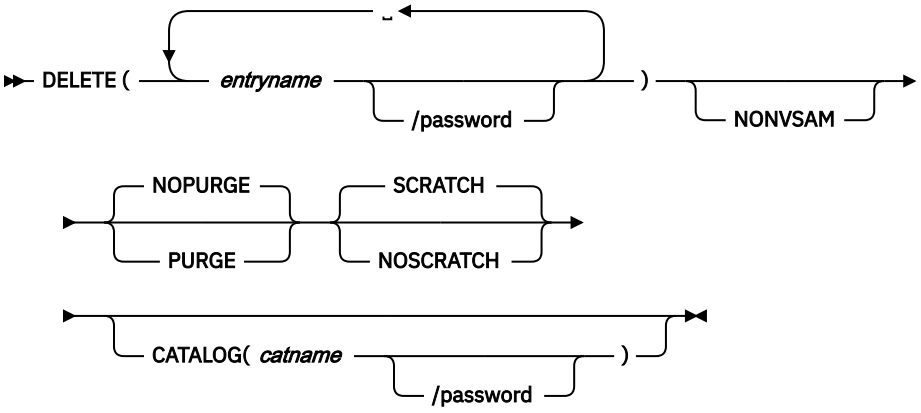
Format 3: Deleting a Cluster or an Alternate Index



Format 4: Deleting a Path



Format 5: Deleting a NonVSAM Object



DELETE Parameters: Summary

The parameters for the DELETE command can be divided into the following categories:

Name, which specifies:

- The entry to be deleted (entryname). This parameter is required.
- The type of entry to be deleted (ALTERNATEINDEX, CLUSTER, MASTERCATALOG, NONVSAM, PATH, SPACE, USERCATALOG).
- The name of the catalog containing the entry to be deleted (CATALOG).

Protection and integrity, which specifies:

- That VSE/VSAM is to overwrite, with binary zeros, the data component of the alternate index or cluster to be deleted (ERASE, NOERASE).
- That VSE/VSAM is to delete nonempty VSE/VSAM data spaces owned by a catalog and give up VSE/VSAM volume use by this catalog for the volumes to be deleted (FORCE, NOFORCE).
- That the object is to be deleted even if its retention period has not expired (PURGE, NOPURGE).
- Whether a nonVSAM file is to have its label removed from the VTOC on the volume on which it resides (SCRATCH, NOSCRATCH).
- That VSE/VSAM try to remove incomplete catalog data that results from a system failure (IGNOREERROR).

DELETE Parameters

(entryname|volser)
names the objects to be deleted (and their catalog entries to be removed), or the volume serial number of the volume that contains data spaces to be deleted.

entryname - is the name of the object (other than a data space) to be deleted. This parameter must be the first one after DELETE. Entry names for up to 100 alternate indexes, clusters, paths, or nonVSAM files can be specified (a list of entry names must be enclosed in parentheses); however, only one catalog name or one volume serial number can be specified at a time. That is, more than one alternate index, cluster, path, or nonVSAM file can be deleted at a time (up to 100), but only one catalog or the data spaces on only one volume can be deleted at a time.

Data and index components cannot be named for deletion. If a component is to be deleted, the name of the cluster or alternate index to which it belongs must be specified. When a cluster is deleted, its components, alternate indexes, and paths are also deleted. When an alternate index is deleted, its components and paths are also deleted, but its base cluster is not deleted.

password - specifies the master password for a password-protected object. The master password may be specified for every entry name or the catalog's master password may be specified through

the CATALOG parameter for the catalog that contains the entries to be deleted. If you are deleting a catalog, the master password must be specified with the entry name.

volser - is the volume serial number of the volume that contains data space(s) to be deleted. Deletion of data space(s) is done by naming the volume on which the data space(s) are defined. If a volume is indicated that contains more than one data space, all empty data spaces owned by the specified catalog are deleted. For details, see the discussion of the SPACE parameter.

If the catalog that defines the data space(s) is password-protected, specify the update or higher-level password in the CATALOG parameter.

ALTERNATEINDEX | CLUSTER | MASTERCATALOG | NONVSAM | PATH | SPACE | USERCATALOG
specifies the type of object to be deleted. Only objects of one type can be deleted in one DELETE if one of these parameters is specified. If one of these parameters is coded and the object named in entryname is not of the specified type, the command is terminated.

ALTERNATEINDEX
specifies that the object to be deleted is an alternate index. It can be deleted only if none of its components are in use by any other partition or system. Deletion of an alternate index causes its data and index components and its related paths to be deleted (its base cluster is unaffected.)

When ERASE is coded, or when the alternate index has the ERASE attribute, the volume containing the alternate index's data component must be mounted. When the alternate index's components reside in unique data spaces, the volumes containing these components must be mounted.

When a unique alternate index is deleted, the alternate index entry and the data and index entries are deleted from the catalog and the data space they occupied is also deleted; however, the volume entries will not be automatically deleted from the catalog. If you want to delete the volume entries, specify SPACE in a separate DELETE command.

Abbreviation: AIX

CLUSTER
specifies that the object to be deleted is a cluster. It can be deleted only if none of its components are in use by any other partition or system. Deletion of a cluster causes its data and index components and its related paths and alternate indexes to be deleted. If you want to overwrite a related alternate index with binary zeros, you must delete it individually with the ERASE parameter before you delete the base cluster.

When ERASE is coded, or when the cluster has the ERASE attribute, the volume containing the cluster's data component must be mounted. When the cluster's components reside in unique data spaces, the volumes containing these components must be mounted.

If a suballocated file flagged notusable is deleted, the space occupied by it will not be freed.

When a unique file is deleted, the cluster entry and the data and index entries are deleted from the catalog and the data space they occupied is also deleted; however, the volume entries are not automatically deleted from the catalog. If you want to delete the volume entries, specify SPACE in a separate DELETE command. The format-1 labels are not removed from the VTOC. However, after removing the VTOC entry of a unique file with the utility IKQVDU, the occupied space is lost for the original catalog and available for new VSE/VSAM space or nonVSAM space.

Abbreviation: CL

MASTERCATALOG
specifies that the object to be deleted is the master catalog. This parameter must be specified to delete the master catalog. It can only be deleted if it is not in use by any other partition (or system) and it is empty, that is, all other objects (except data space entries for the catalog volume) defined in the master catalog (including all user catalogs and the objects defined in them) must have already been deleted. Please also refer to the DISCONNECT parameter of the EXPORT command on page “DISCONNECT ” on page 157.

Abbreviations: MCAT and MRCAT

NONVSAM
specifies that the object to be deleted from the catalog is a nonVSAM file. If the file is on a direct access storage device, its entry will be scratched from the VTOC of the volume(s) on which the file resides unless you specify NOSCRATCH.

Abbreviation: NVSAM

PATH
specifies that the object to be deleted is a path. Deletion of a path does not cause its associated alternate index or base cluster to be deleted.

SPACE
specifies that all empty VSE/VSAM data spaces belonging to one catalog on the volume specified in the volser parameter are to be deleted. SPACE must be specified in a separate DELETE command; it cannot be placed in a list that contains an entry type other than itself.

When all data spaces on the volume that belong to the specified catalog have been deleted and the volume is not in the candidate list of any VSE/VSAM files in this catalog, that volume is no longer available to receive VSE/VSAM files owned by this catalog. If no other catalog owns space on the volume, then the VSE/VSAM ownership flag in the Format 4 record of the VTOC is turned off. To make the volume available once again to the specified catalog, you must define space on the volume in the specified catalog.

If there are nonempty data spaces on the volume or the volume is in a VSE/VSAM file's candidate list, VSE/VSAM ownership of the volume is retained (the volume remains available).

See the FORCE parameter if you want to delete nonempty data spaces or cause the availability of the volume to a catalog to be removed even if the volume is a candidate volume for VSE/VSAM files in this catalog.

Abbreviation: SPC

USERCATALOG
specifies that the object to be deleted is a user catalog entry. This parameter must be specified if you want to delete a user catalog. A user catalog can only be deleted if it is not in use by any other partition (or system) and is empty. That is, all other objects (except data space entries for the catalog volume) defined in the user catalog have already been deleted.

Abbreviation: UCAT

CATALOG(catname)
specifies the name of the catalog that contains the entries to be deleted. CATALOG cannot be specified when a user (UCAT) or master catalog (MCAT) is to be deleted.

If the catalog containing the object to be deleted is password protected, specify the name of the user or master catalog, together with its master password. CATALOG must be specified when nonempty VSE/VSAM data spaces are to be deleted (DELETE SPACE FORCE) and the catalog owning the volume is password-protected.

You do not have to specify CATALOG (unless it is needed for password specification) when the entry to be deleted exists in the default catalog. (The default catalog is (1) the job catalog if one is used, or (2) the master catalog.)

catname - identifies the catalog that contains the entry to be deleted.

password - specifies the password of the catalog. The master password must be specified if (1) objects (other than data spaces and catalogs) which are password-protected are to be deleted and their master passwords were not provided with the entryname parameter, or (2) data spaces owned by a password-protected catalog are to be deleted with the FORCE parameter. The update or higher-level password must be specified if data spaces owned by a password-protected catalog are to be deleted without the FORCE parameter. Otherwise, the password parameter is not required.

Abbreviation: CAT

CLUSTER (see ALTERNATEINDEX)

ERASE | NOERASE
specifies whether the data component of the alternate index or cluster to be deleted is to be erased (overwritten with binary zeros). If specified, this parameter overrides whatever was coded when the alternate index or cluster was defined or last altered.

This parameter cannot be specified for data spaces, catalogs, nonVSAM files, or paths.

ERASE
specifies that the data component will be overwritten with binary zeros. The volume that contains the data component must be mounted.

NOERASE
specifies that the data component will not be overwritten with binary zeros. Therefore, the volume does not have to be mounted.

Abbreviations: ERAS and NERAS

Note: ERASE and NOERASE do not function with spanned record files.

FORCE | NOFORCE
specifies the following:

- Whether nonempty VSE/VSAM data spaces are to be deleted and VSE/VSAM volume use given up
- Whether VSE/VSAM volume use should be given up if the volume is in the candidate list of any VSE/VSAM files
- If the cluster to be deleted is the compression control data set VSAM.COMPRESS.CONTROL, whether the CCDS is to be deleted, even if it is not empty

FORCE
specifies with DELETE SPACE that nonempty VSE/VSAM data spaces belonging to the specified catalog are to be deleted, even if the volume is in the candidate list of any VSE/VSAM files in the catalog. If there are any VSE/VSAM files in the catalog which have space on the volume, their catalog entries are flagged 'not usable' (because owned space has been deleted). If any VSE/VSAM files in the catalog list the volume as a candidate volume, the volume is removed from the candidate list. If the volume is the only volume owned by an unallocated (NOALLOCATION parameter) component, then the catalog entries are flagged 'not usable' (for LISTCAT output).

The command DELETE SPACE FORCE is intended to be a recovery action. It may result in catalog mismatches reported by the utility IKQVCHK which, however, have no effect on catalog consistency. The space occupied by a cluster thus made 'not usable' will not be freed by the command DELETE CLUSTER.

If any of the files in the catalog having space on the volume or having the volume in its candidate list is open, no data spaces are deleted, no files are flagged 'not usable', and the command is rejected.

If you specify FORCE for a volume that contains a VSE/VSAM catalog, the command is rejected. FORCE can only be specified for VSE/VSAM data spaces. The master password of the catalog owning the volume must be specified in the CATALOG parameter if the catalog is password-protected.

If all VSE/VSAM data spaces owned by a catalog are deleted, and no other catalog owns space on the volume, VSE/VSAM volume ownership is removed from the VTOC.

FORCE with DELETE CLUSTER applies to the deletion of the VSE/VSAM compression control data set. It specifies that the compression control data set should be deleted, even if it is not empty. If a nonempty compression control data set is deleted, the access to compressed clusters is no longer possible, because the information how to expand the data is lost.

NOFORCE
specifies that nonempty VSE/VSAM data spaces are not to be deleted, or that a compression control data set should only be deleted if it is empty. VSE/VSAM volume use by the specified catalog is not to be given up if the volume is in the candidate list of any VSE/VSAM files.

Abbreviations: FRC and NFRC

Default: NOFORCE

IGNOREERROR
specifies that VSE/VSAM should try to delete any incomplete catalog information that exists for the named cluster, AIX, or path.

An example of incomplete catalog information is a cluster for which no cluster-level records exists (though data and index words are present). This can happen if a system failure occurs during catalog update for a DEFINE or DELETE command. To correct this situation, specify DELETE entryname CLUSTER IGNOREERROR, where entryname represents the data or index component name. VSE/VSAM will delete that component for you. Then you must redefine the cluster, data, and index components.

If you want to delete a cluster, data, or index component, specify DELETE entryname CLUSTER IGNOREERROR. To delete an alternate index, specify DELETE entryname AIX IGNOREERROR.

When IGNOREERROR processing is complete, VSE/VSAM calls the Catalog Check Service Aid (IKQVCHK) to determine whether the error(s) identified to you have been cleaned up.

For additional explanation to the output, refer to the "Catalog Check Service Aid (IKQVCHK)" in the VSE/VSAM User's Guide and Application Programming.

You should not have to specify IGNOREERROR often. Specify IGNOREERROR only if an error message tells you to. There are two reasons for this:

- Invoking the Catalog Check Service Aid causes performance degradation.
- When catalog information is deleted, it is gone and cannot be retrieved. If you delete the wrong catalog information, you have to rebuild your catalog.

There are some error conditions that will prevent deletion from occurring, even if you specify IGNOREERROR. Some examples are:

- Insufficient storage
- Cluster or AIX already open
- Direct access device space management (DADSM)
- Unable to open catalog
- Unable to find file name in catalog
- Security check failed
- Unable to extract processor ID from the system
- Partition-independent name too long.

Abbreviation: None

MASTERCATALOG (see ALTERNATEINDEX)

NOERASE (see ERASE)

NOFORCE (see FORCE)

NONVSAM (see ALTERNATEINDEX)

NOPURGE (see PURGE)

NOSCRATCH (see SCRATCH)

PATH (see ALTERNATEINDEX)

PURGE | NOPURGE
specifies whether the alternate index, cluster, catalog, nonVSAM file, or path is to be deleted regardless of its retention period.

PURGE
specifies that the object is to be deleted even if its retention period, defined in the TO or FOR parameter, has not expired.

NOPURGE
specifies that the object is not to be deleted if its retention period has not expired.

Abbreviations: PRG and NPRG

Default: NOPURGE

SCRATCH | **NOSCRATCH**
specifies whether a nonVSAM disk file is to be scratched (its label removed from the VTOC) from the volume on which it resides.

SCRATCH
specifies that the nonVSAM disk file being deleted is to have its label(s) removed from the VTOC of the volume(s) on which it resides; that is, both the catalog entry and the file are to be deleted.

NOSCRATCH
specifies that the file being deleted is to keep its label in the VTOC.

Abbreviations: SCR and NSCR

Default: SCRATCH

SPACE (see ALTERNATEINDEX)

USERCATALOG (see ALTERNATEINDEX)

EXPORT

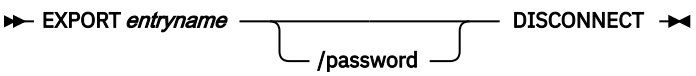
The EXPORT command (EXP) can be used to move clusters, alternate indexes, and user catalogs from one system (or set of systems in a disk sharing environment) to another (with the help of IMPORT), to provide backup copies of clusters and alternate indexes, and to sever the relationship between a user catalog and the master catalog. The EXPORT command cannot be used to move a master catalog, a nonVSAM file, a data space, or a path, or to provide a backup copy of a catalog. However, all paths are automatically exported along with their related base cluster or alternate index. You cannot export an empty cluster or alternate index.

For examples and more information, see:

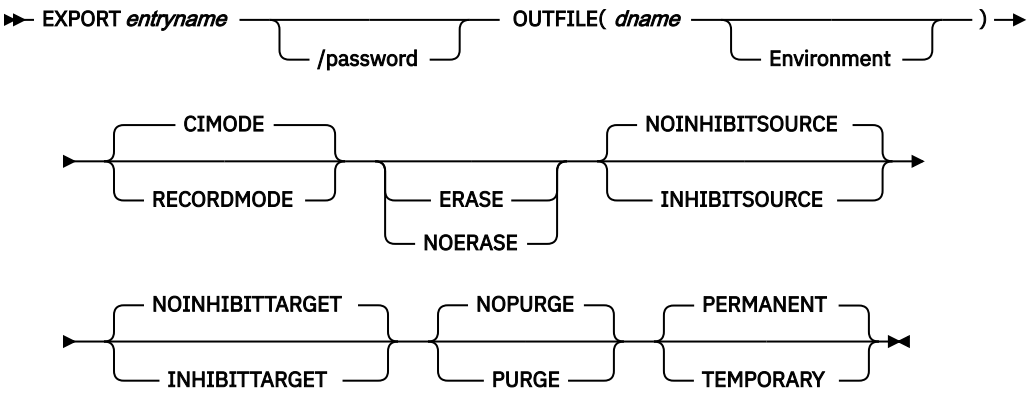
- “Reorganizing a File” on page 42
- “Example 10: Exporting a VSE/VSAM File” on page 227
- “Example 11: Exporting an Alternate Index and Base Cluster” on page 228
- “Example 12: Disconnecting a User Catalog from a Master Catalog” on page 229

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under “PARM” on page 205.)

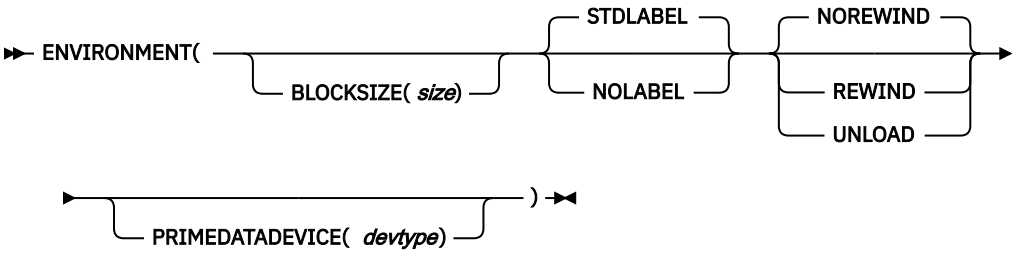
The format of the EXPORT command when it is used to move or disconnect a user catalog is:



The format of the EXPORT command when it is used to move a cluster or alternate index is:



Environment



EXPORT Parameters

entryname

entryname - names the cluster, alternate index, or user catalog to be exported. This parameter must be the first parameter following EXPORT. Data and index components cannot be individually exported; the name of their associated cluster or alternate index must be specified. A path name cannot be used to define a cluster or alternate index.

Entryname (if not a user catalog name) must exist in the default catalog. (The default catalog is (1) the job catalog if one is used, or (2) the master catalog.)

password - if you are exporting a password-protected cluster or alternate index, you must supply the master password of the cluster or alternate index. If any password-protected paths are defined over the cluster or alternate index being exported, you must supply the master password of the master catalog in order to export the protection attribute of all the paths.

If you are exporting a user catalog, you must supply the update or higher-level password of the master catalog, if the master catalog is password-protected.

CIMODE | **RECORDMODE**

specifies whether the data component of a VSE/VSAM file is to be copied to a portable file in control-interval mode or logical-record mode.

RECORDMODE must be specified, and should only be specified, for the conditions explained under RECORDMODE.

CIMODE

specifies that VSE/VSAM CI access is to be used. CI mode enhances performance and is the default, but it does no error checking. The control information in CIDF and RDF must adhere to VSE/VSAM standards. If you write a VSE/VSAM file with MACRF=(CNV) in the ACB, you are responsible for setting up the correct control information.

RECORDMODE

specifies that VSE/VSAM logical record access is to be used.

You *must* specify RECORDMODE if the data CI size is 32K, or if you are exporting:

- A VRDS

- A compressed cluster
- To a z/OS or VSE/ESA system

RECORDMODE should only be used for the above listed conditions.

Abbreviations: CIM and RECM

Default: CIMODE

DISCONNECT
must be specified if a user catalog is to be exported. The entry for the user catalog will be deleted from this system's master catalog. If other VSE/VSAM systems are sharing this master catalog, the user catalog is disconnected from those systems also. The volume that contains the user catalog and any volume on which space is defined for the user catalog must be moved to the system (or systems) to which the catalog will be imported. To make a user catalog available in other systems and in the original system, do not export it, but code the IMPORT CONNECT command to import it to every system in which it is to be available.

A user catalog cannot be disconnected if it is currently open (the catalog or one of its files is processed by another program in this system or a disk sharing system).

Abbreviation: DCON

ERASE | **NOERASE**
specifies whether the data component of the cluster or alternate index to be exported is to be erased (overwritten with binary zeros). If specified, this parameter overrides whatever was specified when the cluster or alternate index was defined or last altered. ERASE or NOERASE can be specified only if the file is to be permanently exported (that is, deleted from the catalog). Note that if you permanently export a cluster, associated alternate indexes (which are implicitly deleted) are not erased even if you specify ERASE (or ERASE was specified when the alternate indexes were defined or last altered). One of the ways to cause alternate indexes to be erased is to permanently export (or explicitly delete) them before permanently exporting the base cluster.

Abbreviations: ERAS and NERAS

INHIBITSOURCE | **NOINHIBITSOURCE**
specifies whether the original cluster or alternate index can be updated. This specification can later be altered with the ALTER command (see INHIBIT, UNINHIBITED).

INHIBITSOURCE
specifies that the original cluster or alternate index can only be read.

The data and index entries in the catalog are modified to indicate update inhibited. When the file is re-imported, the inhibit update flags are reset (unless the copy to be imported was created with the INHIBITTARGET specification).

If INHIBITSOURCE is specified, TEMPORARY must be specified.

NOINHIBITSOURCE
specifies that the original cluster or alternate index can be accessed for any kind of operation and is not to be modified to “inhibit update.” The catalog inhibit indicators are not reset.

Abbreviations: INHS and NINHS

Default: NOINHIBITSOURCE

INHIBITTARGET | **NOINHIBITTARGET**
specifies whether the exported copy of the file can be updated after it has been imported. This specification can later be altered with the ALTER command (see INHIBIT, UNINHIBIT).

INHIBITTARGET
specifies that the exported cluster or alternate index is to be read-only (cannot be updated) after it has been imported into the receiving system. The imported catalog's data and index entries are flagged inhibit update. This imported version of the file will remain flagged “inhibit update” until its components are reset using the UNINHIBIT parameter of the ALTER command.

NOINHIBITTARGET
specifies that the exported copy of the file can be accessed for any type of operation after it has been imported and is not to be modified to “inhibit update.”

Abbreviations: INHT and NINHT

Default: NOINHIBITTARGET

NOERASE (see ERASE)

NOINHIBITSOURCE (see INHIBITSOURCE)

NOINHIBITTARGET (see INHIBITARGET)

NOPURGE (see PURGE)

OUTFILE(dname)
dname specifies the file name of the DLBL or TLBL job control statement that identifies the output file (portable file) to be created as a result of the EXPORT command. This is a required parameter for exporting files.

The target of an EXPORT command cannot be another VSE/VSAM file.

Portable files created by EXPORT must be BAM files on tape or disk devices. For portable files, VSAM is an invalid data set organization. Also, you cannot export multiple files into the same OUTFILE.

If an unlabeled tape is used (NOLABEL option), a TLBL statement is not required. However, because dname is a required positional parameter, a dummy (fictitious) file name of your choosing must be specified.

Abbreviation: OFILE

ENVIRONMENT(subparameters)
describes the parameters of the portable copy.

Abbreviation: ENV

BLOCKSIZE(size)
specifies the block size of the portable copy. For CIMODE, it is recommended that this value be at least eight bytes larger than the file's data component CI size (for enhanced performance). Use RECORDMODE if the data CI size is 32K.

Abbreviation: BLKSZ

Default: 2,048 bytes per block

The maximum block size is:

1. 32,767 bytes per block for tape file and 32,768 bytes per block for sequential FBA disk files.
2. At least 32,768 bytes per block for sequential CKD disk files up to full track size, but not more than 65,535 bytes per block (64K - 1).

The minimum block size is 18 bytes.

NOLABEL | **STDLABEL**
specifies the type of tape (unlabeled or EBCDIC standard-labeled) which contains the portable file, if PRIMEDATADEVICE specifies 2400.

NOLABEL
indicates an unlabeled tape is to be processed. You must properly position the tape to the file you want to process. For an output file, IDCAMS does not write a tape mark as the first record. Instead, a data record is written immediately, wherever the tape is positioned. If an output file is to begin somewhere in the middle of the tape, it is your responsibility to position the tape immediately past the ending tape mark of the preceding file. Use the MTC job control statement or command (described in z/VSE System Control Statements) to properly position the tape or to write a tape mark.

If *VSE/Access Control-Logging and Reporting* (IBM VSE/ACLR) is installed, REWIND must be specified for unlabeled tapes.

Abbreviation: NLBL

Restriction:
If a portable-copy tape file is unlabeled, it must be contained on a single volume, because IMPORT cannot handle a multivolume unlabeled file.

STDLABEL
indicates EBCDIC standard-labeled tapes are to be processed.

Abbreviation: SLBL

Default: STDLABEL

Restriction:
IDCAMS does not support ASCII files or non-standard labels.

NOREWIND | **REWIND** | **UNLOAD**
specifies the tape positioning action for an OPEN, CLOSE, and EOV (end-of-volume) condition, if PRIMEDATADEVICE specifies 2400.

Note: Be aware, when using the REWIND or UNLOAD options (especially with multifile volumes), that any OPEN causes the tape to be positioned at load point. You must be sure that the file you want to process is really the first one (at load point) on the tape. Otherwise, specify the NOREWIND option and use the MTC job control statement (described in z/VSE System Control Statements), for example, to properly position the tape.

NOREWIND
specifies that rewind is never to be performed on OPEN, CLOSE, and EOV.

End-of-file considerations: For EBCDIC standard-labeled tapes, the tape is positioned between the two trailing tape marks. For unlabeled tapes, the tape is positioned following the single trailing tape mark.

Abbreviation: NREW

Default: NOREWIND

REWIND
specifies that tapes are to be rewound to load point but not unloaded on an OPEN, CLOSE, or EOV condition.

Abbreviation: REW

UNLOAD
specifies that tapes are rewound on an OPEN, and rewound and unloaded on an EOV and CLOSE condition.

Abbreviation: UNLD

PRIMEDATADEVICE(devtype)
specify this parameter only if the portable copy resides on a tape device (in which case you specify 2400). The 2400 identifies all tape devices that DTFMT supports. You must specify SYS005 in the ASSGN statement for a tape output file. For more information on tape, see “Tape Considerations for IDCAMS” on page 12.

Abbreviation: PDEV

PERMANENT (see TEMPORARY)

PURGE | **NOPURGE**
specifies whether the cluster or alternate index (not a user catalog) to be permanently exported is to be deleted from the original system(s) regardless of the retention period specified in the TO or FOR parameter when the cluster or alternate index was defined. PURGE or NOPURGE can be specified only if the file is to be permanently exported (that is, deleted from the original system(s)).

PURGE
specifies that the cluster or alternate index is to be deleted even if the retention period has not expired.

NOPURGE
specifies that the cluster or alternate index is not to be deleted if the retention period has not expired.

Abbreviations: PRG and NPRG

Default: NOPURGE

TEMPORARY | **PERMANENT**
specifies whether the cluster or alternate index to be exported is to be deleted from the original system(s).

TEMPORARY
specifies that the cluster or alternate index is not to be deleted from the original system(s). The cluster or alternate index in the original system(s) is marked as temporary to indicate that another copy exists and that the original copy can be replaced (using IMPORT).

PERMANENT
specifies that the cluster or alternate index is to be deleted from the original system(s). The catalog entry is deleted and the storage space used by the cluster or alternate index is freed. If a cluster or alternate index is to be deleted from the original system(s) and its retention period has not expired, PURGE must be coded. When a base cluster is exported, its components, alternate indexes, and paths are also deleted. When an alternate index is exported, its components and path are also deleted, but its base cluster is not deleted.

Abbreviations: TEMP and PERM

Default: PERMANENT

IMPORT

The IMPORT command (IMP) is used in conjunction with the EXPORT command to move clusters, alternate indexes and user catalogs from one system (or set of systems in a disk sharing environment) to another, to restore backup copies of clusters and alternate indexes, and to connect user catalogs to the master catalog. In the process of moving an entry, you may choose to change some of its attributes.

IMPORT will either use a predefined empty file or define the file. If you want the imported file to be in compressed format, you need to define the cluster with the COMPRESSED attribute prior to the IMPORT command.

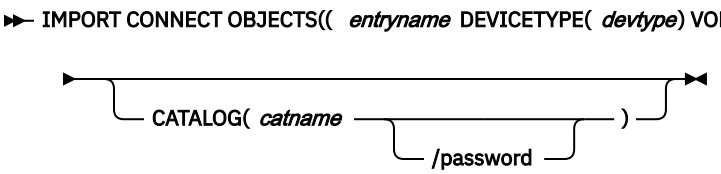
The IMPORT command cannot be used to move a master catalog, a nonVSAM file, a data space or a path. However, all paths exported with their related object are automatically redefined.

For examples and more information, see:

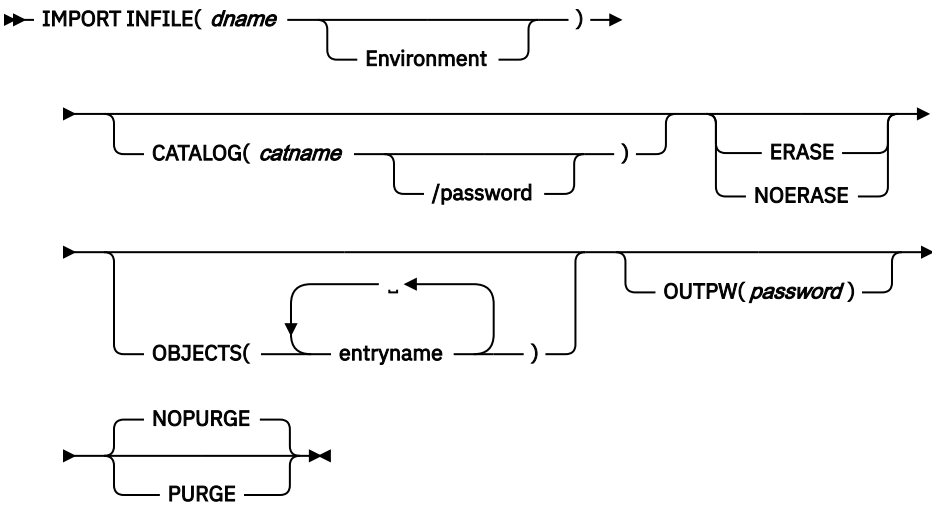
- “Reorganizing a File” on page 42
- “Example 13: Importing a Base Cluster and Alternate Index” on page 229
- “Example 14: Importing an Entry-Sequenced File” on page 230
- “Example 15: Connecting a User Catalog to the Master Catalog” on page 231

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under “PARM” on page 205.)

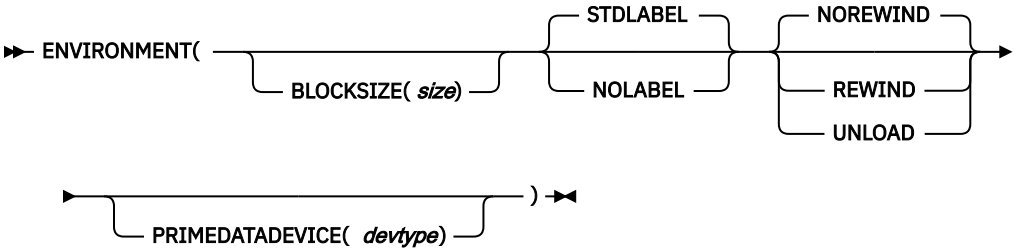
The format of the IMPORT command, when it is used to move or connect a user catalog is:



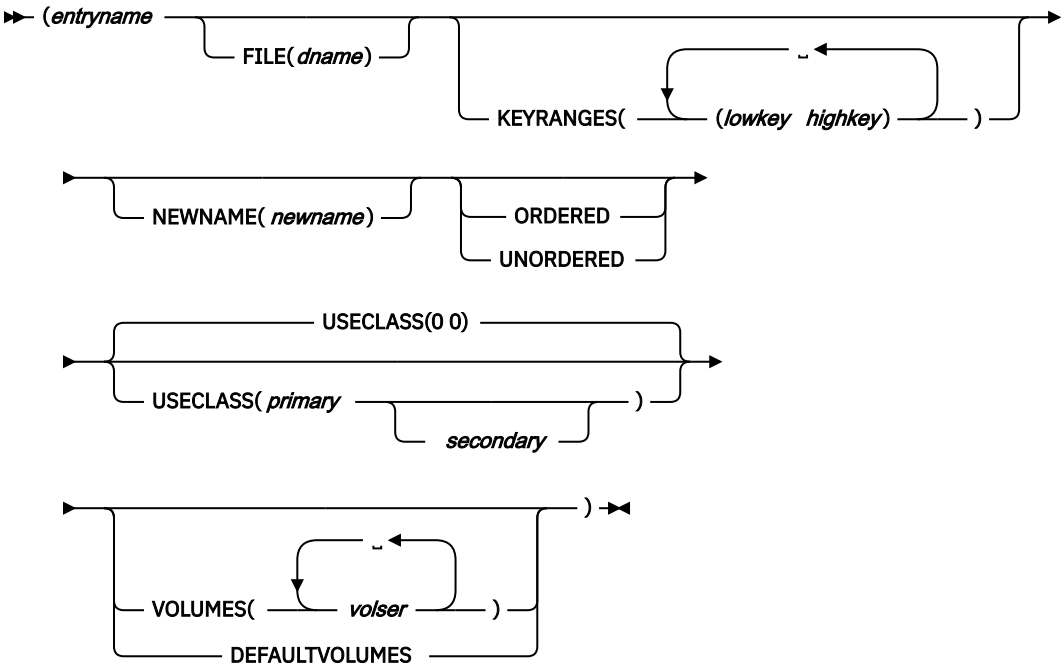
The format of the IMPORT command, when it is used to move or restore a cluster or alternate index, is:



Environment



entryname



IMPORT Parameters

CATALOG(catname)

specifies the name and the password of the catalog in which the imported cluster or alternate index is to be defined or to which the imported user catalogs are to be connected. This parameter is required only when the catalog is password-protected or when you want to direct the imported object to a

catalog other than the default catalog. (The default catalog is (1) the job catalog if one is used, or (2) the master catalog.)

catname - is the name of the catalog.

password - specifies the update or higher-level password of the catalog. If the object being imported is an alternate index over a password-protected base cluster, you should specify the catalog's master password to avoid being prompted for the base cluster's master password.

Abbreviation: CAT

CONNECT
specifies that one or more user catalogs are to be imported. User catalogs are connected to the master catalog in the receiving system(s).

CONNECT must be specified when user catalogs are to be imported. OBJECTS must also be specified to name the catalogs and to identify the volume serial number and device type of the volume that contains them.

In a disk sharing environment where the master catalog is not shared, use IMPORT CONNECT to connect newly-defined user catalogs to the master catalogs in the other systems which are to share them. If the user catalog contains space definitions for one or more other volumes, ensure that such volume or volumes are also accessible by the new system. If not accessible, a volume with the same volume identifier might be overwritten.

Abbreviation: CON

ERASE | NOERASE
specifies whether the data component of a previously temporarily exported cluster or alternate index is to be erased (that is, overwritten with binary zeros) when replaced by the cluster or alternate index to be imported. The object to be imported has the same name as the object it is replacing. The erase operation is actually performed by VSE/VSAM. This parameter overrides whatever was specified when the cluster or alternate index was defined or last altered.

ERASE or NOERASE is specified only if the cluster or alternate index is being imported back into the catalog from which it was temporarily exported.

Note that if you import a cluster causing the old temporarily exported cluster to be deleted, any alternate indexes associated with the old cluster will be implicitly deleted (but not erased). One of the ways to erase the alternate indexes is to explicitly erase them with DELETE before importing the base cluster.

Abbreviations: ERAS and NERAS

INFILE(dname)
specifies the file name of the DLBL or TLBL statement that identifies and describes the portable copy of the cluster or alternate index to be imported. This copy is used to create the new version of the cluster or alternate index in the receiving system. This is a required parameter for importing files. If an unlabeled tape is used (NOLABEL option), a TLBL statement is not required. However, because dname is a required positional parameter, a dummy (fictitious) file name of your choosing must be specified.

Abbreviation: IFILE

ENVIRONMENT(subparameters)
describes the parameters of the portable copy.

Abbreviation: ENV

BLOCKSIZE(size)
specifies the block size of the portable copy created by EXPORT.

Abbreviation: BLKSZ

Default: 2,048 bytes per block.

This default can be used only if the portable copy was created (EXPORT) with the default value.

NOLABEL | STDLABEL
specifies the type of tape (unlabeled or EBCDIC standard-labeled) to contain the portable file, if PRIMEDATADEVICE specifies 2400.

NOLABEL
indicates an unlabeled tape is to be processed. Because the first record of an unlabeled tape may or may not be preceded by a tape mark (user's prerogative), you must take care to properly position the tape to the file you want to process. If a tape mark precedes the file, position the tape either immediately past or immediately before the tape mark. If no tape mark is present, position the tape immediately before the first data record. If the unlabeled tape was created by VSE/VSAM EXPORT there is no tape mark preceding the first record of the file.

If *VSE/Access Control-Logging and Reporting* (IBM VSE/ACLR) is installed, REWIND must be specified for unlabeled tapes.

Abbreviation: NLBL

Restriction: If a portable-copy tape file is unlabeled, it must be contained on a single volume, because IMPORT cannot handle a multivolume unlabeled file.

STDLABEL
indicates EBCDIC standard-labeled tapes are to be processed.

Abbreviation: SLBL

Default: STDLABEL

Restriction:
IDCAMS does not support ASCII files or nonstandard labels.

NOREWIND | REWIND | UNLOAD
specifies the tape positioning action for an OPEN, CLOSE, and EOV (end-of-volume) condition if PRIMEDATADEVICE specifies 2400.

Note: Be aware, when using the REWIND or UNLOAD options (especially with multifile volumes), that any OPEN causes the tape to be positioned at load point. You must be sure that the file you want to process is really the first one (at load point) on the tape. Otherwise, specify the NOREWIND option and use the MTC job control statement (described in z/VSE System Control Statements), for example, to properly position the tape.

NOREWIND
specifies that rewind is never to be performed on OPEN, CLOSE, and EOV.

End-of-file considerations: For EBCDIC standard-labeled tapes, the tape is positioned between the two trailing tape marks. For unlabeled tapes, the tape is positioned following the single trailing tape mark.

Abbreviation: NREW

Default: NOREWIND

REWIND
specifies that tapes are to be rewound to load point but not unloaded on an OPEN, CLOSE, or EOV condition.

Abbreviation: REW

UNLOAD
specifies that tapes are to be rewound on an OPEN, and rewound and unloaded on an EOV and CLOSE condition.

Abbreviation: UNLD

PRIMEDATADEVICE(devtype)
specify this parameter only if the file resides on a tape device (in which case you specify 2400). The 2400 identifies all tape devices that DTFMT supports. You must specify SYS004 in

the ASSGN statement for a tape input file. For more information, see "Tape Considerations for IDCAMS" on page 12.

Abbreviation: PDEV

NOERASE (see ERASE)

NOPURGE (see PURGE)

- For a user catalog:

OBJECTS((entryname

DEVICETYPE(devtype)

VOLUMES(volser)))

- For a cluster or alternate index:

OBJECTS(entryname)

specifies attributes for the cluster, alternate index, their data and index components, or the path or user catalog to be imported. Attributes may be specified for multiple objects by repeating the parameter list for every object.

By repeating the OBJECTS parameter set (maximum of 20 times) for every component and including VOLUMES in every parameter set, you can have the data and index components on different volumes. Although the index and data components may reside on different device types, every volume of a multivolume component must be of the same type.

If the receiving volume is of a device type different from that originally containing the cluster, alternate index, or user catalog, the job may be terminated because of allocation problems due to different cylinder and track capacities. See "Reorganizing a File" on page 42 for techniques to ensure device independence.

entryname - specifies the name of the data component, index component, cluster, alternate index, path, or user catalog whose attributes are being specified.

Abbreviation: OBJ

DEVICETYPE(devtype)

DEVTYPE specifies any supported disk device of the volume that contains the user catalog to be imported (for a list of supported devices, refer to the z/VSE System Control Statements).

- For CKD and ECKD disk devices specify the IBM device number such as 3380 or 3390.
- For FBA devices (including virtual disks) specify FBA.

Abbreviation: DEVT

FILE(dname)

specifies the file name of the DLBL statement that, together with associated EXTENT statement(s), identifies the volume(s) and extents allocated to the data and index components of an alternate index or cluster that is to be imported.

This parameter is required only when the data or index components (of an alternate index or cluster) are defined with the UNIQUE attribute. (If only one component is unique you code one FILE parameter; if both components are unique you code two FILE parameters.)

The symbolic unit can be omitted from the EXTENT statement.

Do not specify the FILE parameter when importing into a predefined empty file.

KEYRANGES(lowkey highkey)

specifies portions of key-sequenced data to be placed on separate volumes. The data is divided by key among the volumes specified in VOLUMES. If a volume serial number was duplicated in VOLUMES, multiple key ranges (of a suballocated object) are placed on that volume. If the number of volumes is greater than the number of key ranges, the excess volumes are used for overflow records from any key range without consideration of range boundaries. If there are fewer volumes than key ranges, the excess key ranges are placed on the last volume specified. The maximum

number of low-key/high-key pairs is 20. Key ranges may not overlap and must be in ascending order; gaps may exist within a specified set of ranges, but records cannot be inserted within a gap.

Every subparameter can contain 1 to 64 characters. All EBCDIC characters are allowed. Subparameters (keys) must be enclosed in single quotation marks if they contain commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a key must be coded as two single quotation marks if the key is enclosed in single quotation marks. If the key is specified in hexadecimal, it must be coded (X‘key’).

lowkey - specifies the low key of the key range. If lowkey is shorter than the actual keys, it will be padded on the right with binary zeros. However, the low key must be of the same length in all key ranges.

highkey - specifies the high key of the key range. If highkey is shorter than the actual keys, it will be padded on the right with binary ones. However, the high key must be of the same length in all key ranges.

Restriction:

KEYRANGES may be specified only with key-sequenced clusters or alternate indexes or their data components. If KEYRANGES is specified for a unique object, every key range must reside on a separate volume.

For more information on keys and KEYRANGES, refer to the "Multiple Volume Support" in the VSE/VSAM User's Guide and Application Programming.

Abbreviation: KRNG

NEWNAME(newname)

specifies the new name of an imported cluster or alternate index or its components, or of a path. You cannot change the name of an imported catalog. The new name may contain 1 to 44 alphameric, national (@, #, and $), and special (hyphen and 12-0 overpunch) characters. Names that contain more than eight characters must be segmented by periods; one through eight characters may be specified between periods. The first character of the name or name segment must be either an alphabetic character or a national character. The last character in the name must not be a period.

If you are importing a cluster or alternate index into a predefined empty file with a different file ID, you must use the NEWNAME parameter to rename the file ID of the cluster or alternate index to match the file ID of the predefined empty file.

If you are importing an object back into the catalog from which it was exported with the TEMPORARY option, you must rename the object and each of its components and paths unless (a) you specify PURGE to delete the old copy, or (b) the old copy has reached its expiration date or was defined without an expiration date.

Abbreviation: NEWNM

ORDERED | **UNORDERED**

specifies whether volumes are to be used in the order in which they were listed in the VOLUMES parameter. If KEYRANGES is also specified, all of the records within the range specified by the first low-key high-key pair are placed on the first volume specified in VOLUMES; all of the records within the second range are placed on the second volume, and so forth. If it is impossible to allocate volumes in the given order and ORDERED is specified, the command is terminated.

Abbreviations: ORD and UNORD

USECLASS(primary)

specifies that the class of an imported object (cluster, alternate index, and their data and index components) is to be modified.

The USECLASS specifications for a cluster and alternate index are modified in the following order:

cluster or alternate index,
data component, and
index component.

For example, if you specify a USECLASS value for both the data component and cluster (in either order) of an imported object:

(1) The value specified for the cluster level is also propagated to the data component level and (if present) index component level,

(2) The value specified for the data component will be inserted into the data component, replacing the previous data component value derived from the cluster specification,

(3) The index component (if present) would retain the propagated cluster USECLASS value.

If you specify only a primary USECLASS value, the default secondary value ('P' - same as primary) is applied to the imported object. This effectively overrides the secondary value for that imported object in the portable file.

If you specify primary USECLASS values other than 0 for a unique cluster or alternate index an error occurs.

Abbreviation: USCL

VOLUMES(volser)|DEFAULTVOLUMES

specifies the volume on which the user catalog resides for IMPORT CONNECT. Only one volume (the one on which the catalog itself resides) may be specified. Ensure that you specify the correct volser because VSE/VSAM does not require that the volume be mounted.

Specifies the volumes on which the cluster, alternate index, data, component, or index component is to reside.

VOLUMES

If the original volume(s) from which the cluster or alternate index was exported is to be the receiving volume(s), VOLUMES is not required. If VOLUMES is specified at the cluster or alternate index level, it will propagate to the data and index levels. A data or index level specification will override a cluster or alternate-index level specification.

volser - contains one through six alphameric, national (@, #, and $), hyphens, and special characters: commas, semicolons, blanks, parentheses, slashes, asterisks, periods, quotation marks, ampersands, plus signs, and equal signs. The volume serial numbers must be enclosed in single quotation marks if they contain special characters. Single quotation marks within a volume serial number must be coded as two single quotation marks. For consistency with z/VSE job control, only alphameric characters should be used.

VOLUMES must be specified when a user catalog is to be imported (only one volume). VOLUMES or DEFAULTVOLUMES must be specified when the object was permanently exported from:

- DOS/VS Release 34 or earlier
- MVS™ Release 3.1 or earlier
- OS/VS2 Release 3 or earlier.

All volumes for a component of a cluster or alternate index must be of the same device type.

DEFAULTVOLUMES

indicates that (for the specified cluster, alternate index, data or index component) VSE/VSAM is to disregard the exported volume list(s) and, instead, is to obtain a new list of volume(s). VSE/VSAM obtains the new volume(s) list from the data and index (if present) components of the applicable default models in the receiving catalog. The default model used corresponds to the type of file being imported (alternate index, key-sequenced file, etc.).

An error message is issued if you specify DEFAULTVOLUMES and VOLUMES together in the same entryname list. You can specify DEFAULTVOLUMES at any level and in any combination of levels; if specified at the alternate index or cluster level, DEFAULTVOLUMES propagates to the data component level and (if applicable) the index component level. (If DEFAULTVOLUMES is propagated to the data or index component level where the VOLUMES parameter has been specified, the VOLUMES parameter takes precedence.)

DEFAULTVOLUMES cannot be specified for components which are defined with the ORDERED or UNIQUE parameters.

Abbreviations: DFVOL and VOL

Default: If neither VOLUMES nor DEFAULTVOLUMES is specified, the original volume(s) from which the object was exported is used.

OUTPW(password)
specifies the password of the file to receive the portable copy.

password - is required only if importing into a predefined empty file that has passwords that are different from the old passwords of the file being imported. You must specify the file's update or higher-level password.

Abbreviation: OPW

PURGE | NOPURGE
specifies whether the entry for the original cluster or alternate index is to be deleted and replaced by a new entry with the same name regardless of the retention time specified in the TO or FOR parameter in the original definition.

PURGE
specifies that the entry for the original cluster or alternate index is to be deleted even if the retention period has not expired. This parameter can be used only when you are importing the cluster or alternate index into the original catalog from which it was exported with the TEMPORARY parameter.

NOPURGE
specifies that the entry for the original cluster or alternate index is not to be deleted if the retention period has not expired. If you specify NOPURGE and the retention period has not expired, you get a message that tells you that the cluster or alternate index has not been deleted and the command is terminated. The portable copy of the object that was to have been imported is still usable.

Abbreviations: PRG and NPRG

Default: NOPURGE

LISTCAT

The LISTCAT command (LISTC) is used to list entries contained in a catalog. The ENTRIES parameter is used to specify the name(s) of individual entries to be listed. The AIX, CL, DATA, INDEX, NONVSAM, PATH, SPACE, and UCAT parameters are used to specify the type(s) of entries to be listed. You can specify as many entry types as you want. If you want to completely list a catalog, do not specify any entry type. The NAME, VOLUME, ALLOCATION, and ALL parameters specify the fields to be listed for every catalog entry.

Table 10 on page 167 shows the kinds of listings that result from coding various combinations of entryname and type of object.

Table 10. Listing Contents by Parameters

Requested type is:	**CL**	**AIX**	**DATA**	**IX**	**PATH**	**SPC**	**NVSAM**	**UCAT**	**None**
Entryname type is:									
CL	L*	E	A*	A*	A*	E	E	E	LA*
AIX	E	L*	A*	A*	A*	E	E	E	LA*
DATA	E	E	L*	E	E	E	E	E	L*
IX	E	E	E	L*	E	E	E	E	L*

Table 10. Listing Contents by Parameters (continued)

Requested type is:	CL	AIX	DATA	IX	PATH	SPC	NVSAM	UCAT	None
PATH	E	E	E	E	L*	E	E	E	L*
SPC	U	U	U	U	U	L*	U	U	U
NVSAM	E	E	E	E	E	E	L*	E	L*
UCAT	MU	E	E	E	E	E	E	L*	UM

Explanation for Table 10 on page 167:

```
AIX    = alternate index
CL     = cluster
IX     = index
NVSAM  = non VSAM
SPC    = space
UCAT   = user catalog

* shows which combinations produce a useful listing
A  = list the desired association
E  = error message: "not a requested type"
L  = list the entry
LA = list the entry and associated entries
MU = if listing from master catalog, "E"
     if listing from user catalog, "L"
U  = unpredictable results
UM = if listing from master catalog, "L"
     if listing from user catalog, "LA"
```

Note: If any of the above entryname types are not in the catalog, IDCAMS generates an appropriate information message.

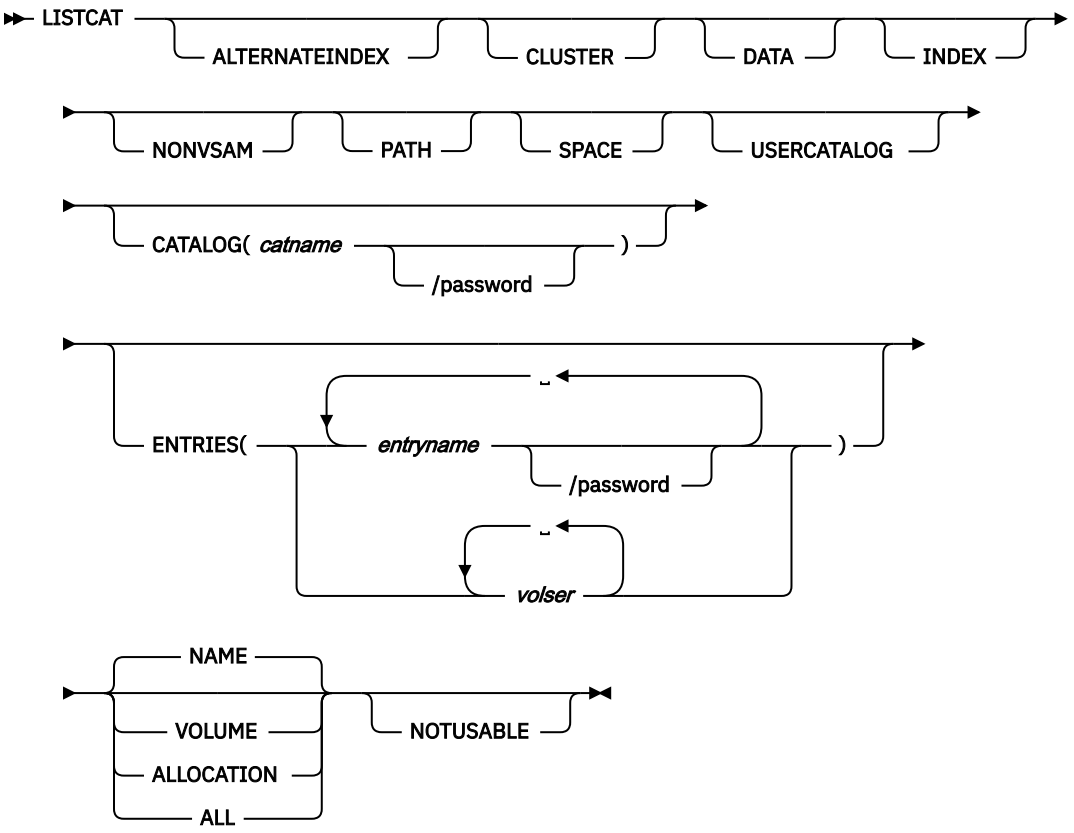
For more information and examples, see:

- "Listing Catalog Entries" on page 47
- Appendix A, "Interpreting LISTCAT Output," on page 271
- Examples:
  - "Example 3: Define VSE/VSAM Files" on page 212
  - "Example 4: Define NonVSAM and VSE/VSAM Files" on page 215
  - "Example 7: Modifying and Listing the Cataloged Attributes of a File" on page 222

Refer also to LISTCAT in the index of the VSE/VSAM User's Guide and Application Programming.

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (See the description of SYNCHK parameter of "PARM" on page 205 for further information.)

The format of the LISTCAT command is:



LISTCAT Parameters

ALTERNATEINDEX
specifies that alternate index entries are to be listed. If ALTERNATEINDEX is specified but DATA, INDEX, and PATH are not specified, entries for the alternate index's data and index components and path(s) are not listed.

Abbreviation: AIX

CATALOG(catname)
specifies the catalog that contains the entries that are to be listed. This parameter is required only when the catalog is password-protected or the entry to be listed is in a catalog other than the default catalog. (The default catalog is (1) the job catalog if one is used, or (2) the master catalog.)

catname - is the name of the catalog.

password - specifies the read or higher-level password of the catalog if it is password-protected. If the entries to be listed contain password protection information and this information is to be listed, then the master password must be supplied either through this parameter or through the ENTRIES parameter.

Abbreviation: CAT

CLUSTER
specifies that cluster entries are to be listed. If CLUSTER is specified but DATA, INDEX, and PATH are not specified, entries for the cluster's data and index components and path are not listed.

Abbreviation: CL

DATA
specifies that entries for data components, excluding the data component of the catalog, are to be listed. If a cluster or alternate index's name is specified under ENTRIES and DATA is coded, only the data component entry is listed.

ENTRIES(entryname) | (volser)
specifies the entries to be listed. When you identify an entry with its name and also specify an entry type, the named entry is only listed if it is of the specified type. If ENTRIES is not specified and no entry-types are specified, the entire catalog is listed. You can also specify the fields you want listed for every entry (ALL, NAME, and so forth). If you want information about a user catalog, the catalog's volume must be physically mounted. You then specify the catalog's name as the entryname. (You cannot specify the catalog's components by name.)

If you want information about the data spaces on particular volumes, specify SPACE and the volumes' serial numbers as volser.

password - specifies a password when the entry to be listed is password protected and a password was not specified through the CATALOG parameter. The password must be the entry's read or higher-level password. If protection attributes are to be listed, you must supply the entry's master password; if no password is supplied, the operator is prompted for every entry's password.

Abbreviations: ENT or ENTRY

INDEX
specifies that entries for index components, excluding the index component of the catalog, are to be listed. If a cluster or alternate index's name is specified under ENTRIES and INDEX is coded, only the index component entry is listed.

Abbreviation: IX

NAME | VOLUME | ALLOCATION | ALL
specifies the fields to be listed for every catalog entry.

NAME
specifies that the name and type of the objects defined by every entry are to be listed. NAME is the default.

VOLUME
specifies that the name of the object defined by every entry, its history information (owner ID, creation and expiration dates, the VSE/VSAM release ID when it was created), its volume serial number, and device type allocated to it are to be listed. Volume information (volser and devtype) is only listed for data and index component entries, user catalogs, and nonVSAM file entries.

Abbreviation: VOL

ALLOCATION
specifies that volume information and detailed information about the allocation are to be listed. This information is listed only for data and index component entries. Types of entries should only specify DATA or INDEX. ENTRIES can specify cluster, alternate index, data or index component names.

Abbreviation: ALLOC

ALL
specifies that all the fields in an entry are to be listed, including protected attributes, if the entry's or the catalog's master password is specified.

Note: If ALL is specified for more than one entry, all fields of all entries are listed. This means that certain fields are listed more than once: for example, the path field using the AIX entry and the path entry.

Default: NAME

NONVSAM
specifies that entries for nonVSAM files are to be listed.

Abbreviation: NVSAM

NOTUSABLE
specifies that only those entries for data and index components that are flagged as notusable are to be listed. They are so flagged because data space has been deleted from the volume by the FORCE option of DELETE SPACE or the last volume of a NOALLOCATION file.

Abbreviation: NUS

PATH
specifies that entries for paths are to be listed.

SPACE
specifies that entries for volumes containing data spaces defined in the catalog are to be listed. Candidate volumes are included. If entries are identified by entryname, SPACE can be coded only when no other entry type is coded. You must specify SPACE if ENTRIES specifies a volser (volume serial number).

Abbreviation: SPC

USERCATALOG
specifies that user catalog pointers in the master catalog are to be listed. USERCATALOG is applicable only when the catalog to be listed is the master catalog.

Abbreviation: UCAT

PRINT

The PRINT command is used to list part or all of an indexed-sequential (ISAM), sequential, or VSE/VSAM file. A base cluster is listed in alternate key sequence using a path.

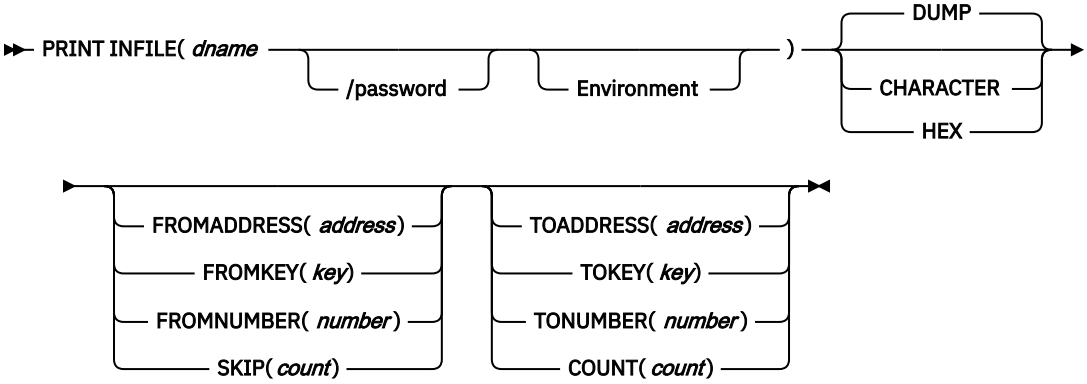
The IDCAMS PRINT command is a basic method of listing records from a VSAM file. VSE/VSAM printing of files is done in upper case characters without respect to, for example, lower case or foreign language special characters, some information may not be readable. This problem may be solved by a user-provided translate table for IDCAMS PRINT which can be provided via the PARM modal command for IDCAMS. Refer to “PARM” on page 205.

For more information and examples, see:

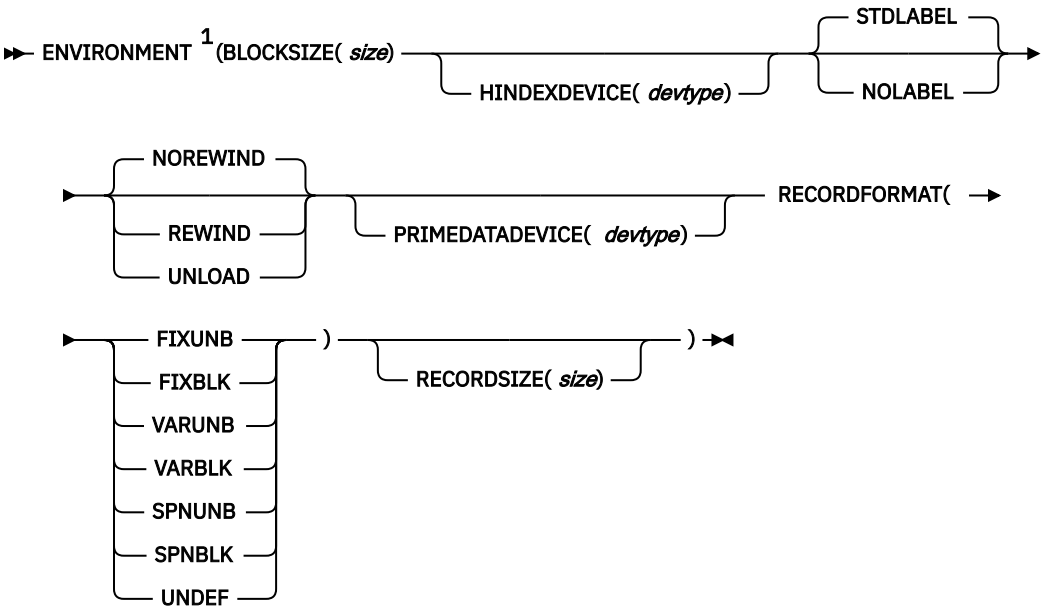
- “Printing Data Records” on page 47
- Figure 7 on page 177 for sample listings produced by the PRINT command.
- “Example 5: Loading and Printing of Files” on page 218
- “Example 6: Modifying and Printing the Contents of VSE/VSAM Files” on page 221
- “Example 8: Creating an Alternate Index and Its Path” on page 223.
- “Example 25: IDCAMS PRINT That Allows Printing in Upper and Lower Case” on page 239

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under “PARM” on page 205.)

The format of the PRINT command is:



Environment



Notes:

[1] This parameter is required under certain conditions.

CHARACTER | DUMP | HEX
specifies the format of the listing (see Figure 7 on page 177).

CHARACTER
specifies that every byte in every record is to be printed as a character. Bit patterns not defining a character are printed as periods. Key fields are listed in character format.

Abbreviation: CHAR

DUMP
specifies that every byte in every record is to be printed in both hexadecimal and character format. In the character portion of the listing, bit patterns not defining a character are printed as periods. Key fields are listed in character and hexadecimal formats.

HEX
specifies that every byte in every record is to be printed as two hexadecimal digits. Key fields are listed in hexadecimal format.

Default: DUMP

ENVIRONMENT(subparameters)
describes the input file if it is a nonVSAM file. When you are printing a nonVSAM file or a SAM ESDS file and the record format, record size, or block size has been changed using DTF access, then this parameter with the BLOCKSIZE and RECORDFORMAT subparameters is required.

Abbreviation: ENV

BLOCKSIZE(size)
specifies the block size for a nonVSAM file. This parameter is required. For FIXBLK files, substitute the logical record length times the number of records per block. For ISAM FIXUNB files, substitute the logical record length plus the key length.

Abbreviation: BLKSZ

HINDEXDEVICE(devtype)
specifies, for an ISAM file, the device type of the volume on which the highest index resides.

The possible device types that apply to an ISAM file are: 2314, 2319, 3330, and 3340.

Abbreviation: HDEV

Default: The device type of the highest index defaults to the device type of the volume on which the prime data records reside (see PRIMEDATADEVICE).

NOLABEL | **STDLABEL**
specifies the type of tape (unlabeled or EBCDIC standard-labeled) to be processed if PRIMEDATADEVICE specifies 2400.

NOLABEL
indicates an unlabeled tape is to be processed. Because the first record of an unlabeled tape may or may not be preceded by a tape mark (user's prerogative), you must take care to properly position the tape to the file you want to process. If a tape mark precedes the file, position the tape either immediately past or immediately before the tape mark. If no tape mark is present, position the tape immediately before the first data record. If the unlabeled tape was created by VSE/VSAM IDCAMS, there is no tape mark preceding the first record of the file.

If *VSE/Access Control-Logging and Reporting* (IBM VSE/ACLR) is installed, REWIND must be specified for unlabeled tapes.

Abbreviation: NLBL

Restriction: PRINT cannot process multivolume unlabeled files.

STDLABEL
indicates EBCDIC standard-labeled tapes are to be processed.

Abbreviation: SLBL

Default: STDLABEL

Restriction: IDCAMS does not support ASCII files or nonstandard labels.

NOREWIND | **REWIND** | **UNLOAD**
specifies the tape positioning action for an OPEN, CLOSE, and EOV (end-of-volume) condition if PRIMEDATADEVICE specifies 2400.

Note: Be aware, when using the REWIND or UNLOAD options (especially with multifile volumes), that any OPEN causes the tape to be positioned at load point. You must be sure that the file you want to process is really the first one (at load point) on the tape. Otherwise, specify the NOREWIND option and use the MTC job control statement (described in z/VSE System Control Statements), for example, to properly position the tape.

NOREWIND
specifies that rewind is never to be performed on OPEN, CLOSE, and EOV.

End-of-file considerations: For EBCDIC standard-labeled tapes, the tape is positioned between the two trailing tape marks. For unlabeled tapes, the tape is positioned following the single trailing tape mark.

Abbreviation: NREW

Default: NOREWIND

REWIND
specifies that tapes are to be rewound to load point but not unloaded on an OPEN, CLOSE, or EOV condition.

Abbreviation: REW

UNLOAD
specifies that tapes are to be rewound on an OPEN, and rewound and unloaded on an EOV and CLOSE condition.

Abbreviation: UNLD

PRIMEDATADEVICE(devtype)
specifies the device type of the volume on which the prime data records reside.

The possible device types that apply to an ISAM file are: 2314, 2319, 3330, and 3340.

You do not have to specify this parameter for a sequential file unless the file resides on tape (in which case you specify 2400). The 2400 identifies all tape devices that DTFMT supports. You must specify SYS004 in the ASSGN statement for a tape input file.

Abbreviation: PDEV

Default: 2314

RECORDFORMAT(format)
specifies the format of the records in the nonVSAM file. This parameter is required. For format, substitute one of the following values:

Value
Meaning

FIXUNB | **F**
Fixed, unblocked

FIXBLK | **FB**
Fixed, blocked

VARUNB | **V**
Variable, unblocked

VARBLK | **VB**
Variable, blocked

SPNUNB | **S**
Variable spanned, unblocked

SPNBLK | **SB**
Variable spanned, blocked

UNDEF | **U**
Undefined

Abbreviation: RECFM

RECORDSIZE(size)
specifies the length of a logical record for FIXBLK or FIXUNB nonVSAM files. For indexed-sequential FIXUNB files, this parameter is the logical record length plus the key length. For UNDEF, SPNBLK, or SPNUNB nonVSAM files, this parameter is the maximum record size. This parameter is not necessary for VARUNB or VARBLK.

Abbreviation: RECSZ

Default: If RECORDSIZE is not specified, blocking is one record per block. Therefore, RECORDSIZE is equal to block size for files that are FIXUNB, FIXBLK, or UNDEF. RECORDSIZE is equal to block size minus four for files that are VARUNB, VARBLK, SPNUNB, or SPNBLK.

FROMADDRESS(address) | FROMKEY(key) | FROMNUMBER(number) | SKIP(count)
specifies the first record to print in the file. For the files and keywords allowed in combination with these parameters, see Table 12 on page 186.

FROMADDRESS(address)
specifies the RBA of the first record. It must be the beginning of a logical record. If you specify this parameter for a key-sequenced file, the listing will be sequenced by RBA rather than by key. The address may be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

FROMKEY(key)
specifies the key of the first record you want listed. It may be a full key or a generic key -- that is, the high-order portion of a key. If you specify a generic key, listing begins at the first record whose key matches the portion of the key you specified. (You must not specify a key longer than that defined for the file. If you do, the listing is not performed.) If the specified key is not found, listing begins with the next higher key. The key must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a key must be coded as two single quotation marks if the key is enclosed in single quotation marks; it may contain up to 255 bytes, that is, 510 hexadecimal or 255 EBCDIC characters. If it is specified in hexadecimal, it must be preceded by X and be enclosed in single quotation marks.

FROMNUMBER(number)
to be printed. Do not enclose the value in quotation marks.

SKIP(count)
specifies the number of records you want to skip before the listing of records begins. For example, if you want the listing to begin with the 500th record, you specify SKIP(499). The count can be expressed in decimal, hexadecimal, or binary. If it is specified in hexadecimal or binary, it must be preceded by X or B, respectively, enclosed in single quotation marks, and no longer than one fullword (4 bytes or 32 bits).

Abbreviations: FADDR, FKEY, and FNUM

Default: If nothing is specified, the listing begins with the first record.

INFILE(dname)
describes the input file. If the file is defined in a user catalog which is not the job catalog, you must explicitly name the user catalog by specifying the CAT parameter in the file's DLBL job control statement. (The user catalog's DLBL statement does not require an associated EXTENT statement.)

dname - specifies the file name of the DLBL (an associated EXTENT statement is required only if the input file is nonVSAM) or TLBL statement that identifies the input file to be listed. This is a required parameter. You can list a base cluster in alternate key sequence by specifying a path name as the file ID, in the DLBL statement. If an unlabeled tape is used (NOLABEL option), a TLBL statement is not required. However, because dname is a required positional parameter, a dummy (fictitious) file name of your choosing must be specified.

password - specifies the read or higher-level password of a password protected VSE/VSAM file or path. For printing a data or index component, supply the password of the cluster or alternate index to which the component belongs; for a path, it is the password of the path.

Abbreviation: IFILE

TOADDRESS(address) | TOKEY(key) | TONUMBER(number) | COUNT(count)
specifies the last record to print in the file. The last record must follow the first record. For the files and keywords allowed in combination with these parameters, see Table 13 on page 188.

TOADDRESS(address)
specifies the RBA of the last record. Unlike FROMADDRESS, the RBA does not need to be the beginning of a logical record. The entire record containing the specified RBA is printed. If you specify this parameter for a key-sequenced file, the listing will be sequenced by RBA rather than by key. The address can be expressed in decimal, hexadecimal, or binary. If it is specified

in hexadecimal or binary, it must be preceded by X or B, respectively, and enclosed in single quotation marks.

TOKEY(key)
specifies the key of the last record. It may be a full key or a generic key - that is, the high-order portion of a key. If you specify a generic key, listing stops after the last record is listed whose key matches the portion of the key you specified. (You must not specify a key longer than that defined for the file. If you do, the listing is not performed.) If the specified key is not found, listing ends with the next lower key. The key must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a key must be coded as two single quotation marks if the key is enclosed in single quotation marks; it may contain up to 255 bytes, that is, 510 hexadecimal or 255 EBCDIC characters. If it is specified in hexadecimal, it must be preceded by X and be enclosed in single quotation marks.

TONUMBER(number)
specifies the relative-record number (decimal) of the last record to be printed. Do not enclose the value in quotation marks.

COUNT(count)
specifies the number of records to be listed. The count can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

Abbreviations: TADDR and TNUM

Default: If nothing is specified, the listing ends with the last record.

Examples of PRINT Command Output

```
// JOB TESTVSAM                                                    DATE 10/11/88,CLOCK 10/14/09
// DLBL UCAT,'TST.UCAT',,VSAM                                     10000050
// DLBL KSDS,'TST.KSDS.CLUSTER',,VSAM,CAT=UCAT                    10000070
// EXEC IDCAMS,SIZE=AUTO                                          10000100


IDCAMS  SYSTEM SERVICES                                  TIME: 10:14:10       10/11/88

 PRINT INFILE(KSDS) CHARACTER  COUNT(3)                        10000110

IDCAMS  SYSTEM SERVICES                                  TIME: 10:14:10       10/11/88
LISTING OF DATA SET -TST.KSDS.CLUSTER
KEY OF RECORD - 00001
00001 +        HELPIPCS G1 F         80         60          2  1/03/85 11:15:06
KEY OF RECORD - 00002
00002 $$BENDQB TEXT     G2 F         80          3          1 12/20/84 13:12:28
KEY OF RECORD - 00003
00003 $$BOMSGY TEXT     G2 F         80         15          1 12/20/84 13:50:21
IDC0005I NUMBER OF RECORDS PROCESSED WAS 3
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDCAMS  SYSTEM SERVICES                                     TIME: 10:14:10       10/11/88

 PRINT INFILE(KSDS) DUMP       COUNT(3)                          100001 20

IDCAMS  SYSTEM SERVICES                                     TIME: 10:14:10       10/11/88     PAGE   4
LISTING OF DATA SET -TST.KSDS.CLUSTER

KEY OF RECORD -
000000  F0F0F0F0 F1                                                          *00001                          *

RECORD -
000000  F0F0F0F0 F1404E40 40404040 404040C8   C5D3D7C9 D7C3E240 C7F140C6 40404040   *00001 +        HELPIPCS G1 F    *
000020  40404040 40F8F040 40404040 40404040   F6F04040 40404040 40404040 F24040F1   *     80         60          2  1*
000040  61F0F361 F8F540F1 F17AF1F5 7AF0F640                                        */03/85 11:15:06                *

KEY OF RECORD -
000000  F0F0F0F0 F2                                                          *00002                          *

RECORD -
000000  F0F0F0F0 F2405B5B C2C5D5C4 D8C240E3   C5E7E340 40404040 C7F240C6 40404040   *00002 $$BENDQB TEXT     G2 F    *
000020  40404040 40F8F040 40404040 40404040   40F34040 40404040 40404040 F140F1F2   *    80          3          1  12*
000040  61F2F061 F8F440F1 F37AF1F2 7AF2F840                                        */20/84 13:12:28                *

KEY OF RECORD -
000000  F0F0F0F0 F3                                                          *00003                          *

RECORD -
000000  F0F0F0F0 F3405B5B C2D6D4E2 C7E840E3   C5E7E340 40404040 C7F240C6 40404040   *00003 $$BOMSGY TEXT     G2 F2   *
000020  40404040 40F8F040 40404040 40404040   F1F54040 40404040 40404040 F140F1F2   *     80          15          1 12*
000040  61F2F061 F8F440F1 F37AF5F0 7AF2F140                                        */20/84 13:50:21                *

IDC0005I NUMBER OF RECORDS PROCESSED WAS 3
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDCAMS  SYSTEM SERVICES                                     TIME: 10:14:10       10/11/88     PAGE   5
```

```
 PRINT INFILE(KSDS) HEX        COUNT(3)                          10000130
IDCAMS  SYSTEM SERVICES                                     TIME: 10:14:10       10/11/88     PAGE   6
LISTING OF DATA SET -TST.KSDS.CLUSTER
KEY OF RECORD - F0F0F0F0F1
F0F0F0F0F1404E4040404040404040C8C5D3D7C9D7C3E240C7F140C64040404040404040404040F8F04040404040404040404040F6F0404040404040404040
F24040F161F0F361F8F540F1F17AF1F57AF0F640
KEY OF RECORD - F0F0F0F0F2
F0F0F0F0F2405B5BC2C5D5C4D8C240E3C5E7E34040404040C7F240C64040404040404040404040F8F04040404040404040404040F3404040404040404040
F140F1F261F2F061F8F440F1F37AF1F27AF2F840
KEY OF RECORD - F0F0F0F0F3
F0F0F0F0F3405B5BC2D6D4E2C7E840E3C5E7E34040404040C7F240C64040404040404040404040F8F04040404040404040404040F1F5404040404040404040
F140F1F261F2F061F8F440F1F37AF5F07AF2F140
IDC0005I NUMBER OF RECORDS PROCESSED WAS 3
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDCAMS  SYSTEM SERVICES                                     TIME: 10:14:10       10/11/88     PAGE   7

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
1S55I  LAST RETURN CODE WAS 0000
EOJ TESTVSAM  MAX.RETURN CODE=0000                                  DATE 10/11/88,CLOCK 10/14/15,DURATION 00/00/05
```

Figure 7. Output Example of PRINT DUMP Command

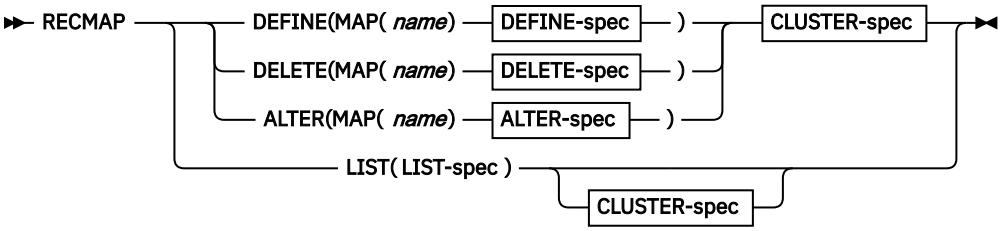
RECMAP

This method of defining a map for a VSAM cluster uses the IDCAMS **RECMAP** command. Using RECMAP, you can also delete, change (alter), or list the contents of a map or view.

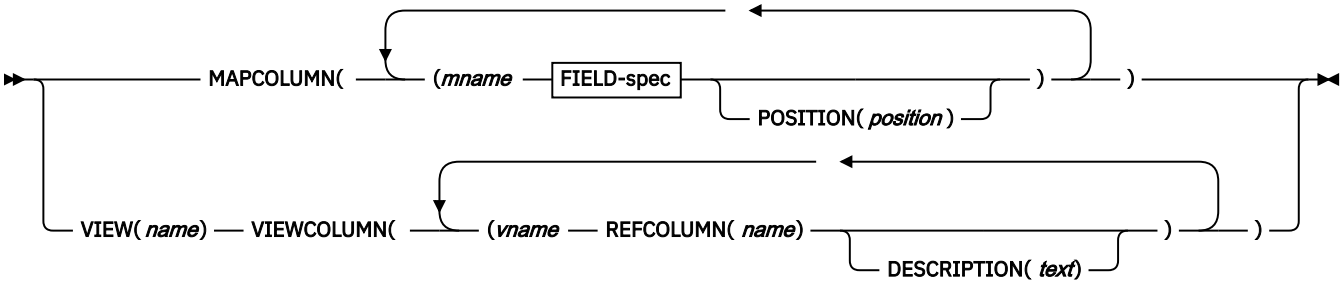
Note: You cannot define more than 255 columns using a single RECMAP command. If you wish to define more than 255 columns, you must repeat the RECMAP command to define additional columns.

For a practical example of how RECMAP can be used, refer to the z/VSE e-business Connectors User's Guide.

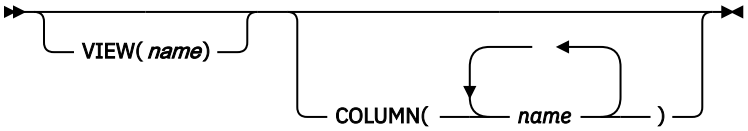
The syntax of this command is:



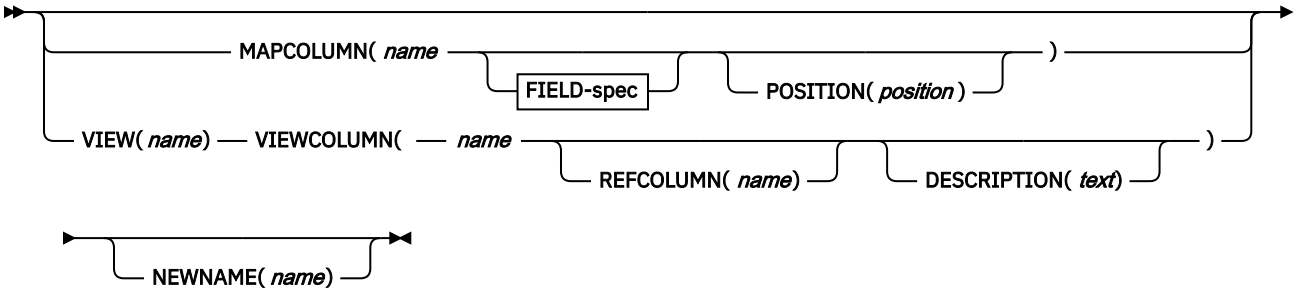
DEFINE-spec



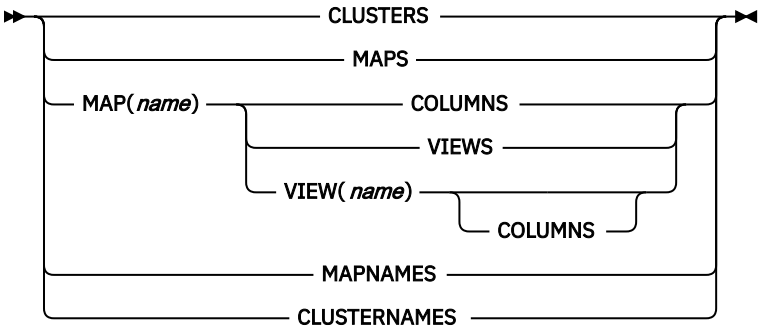
DELETE-spec



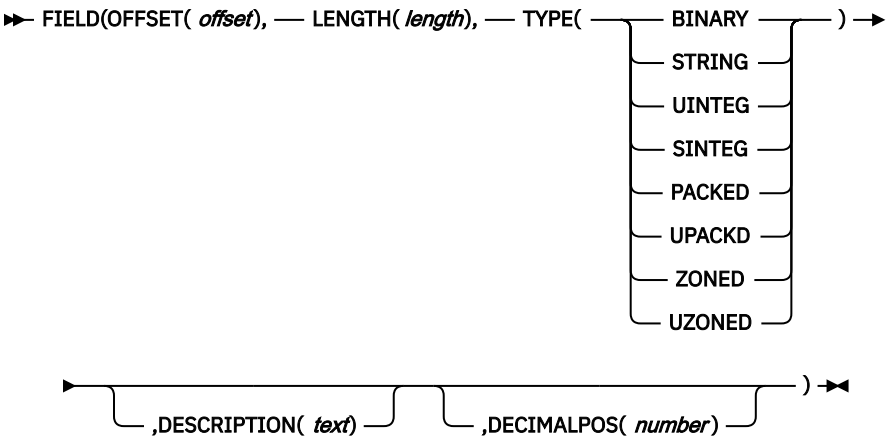
ALTER-spec



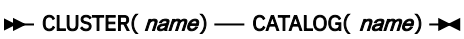
LIST-spec



FIELD-spec



CLUSTER-spec



RECMAP Parameters

CATALOG(name)
The name of a VSAM catalog that you want to reference as was specified in the DEFINE CATALOG job.

Abbreviation: CAT

CLUSTER(name)
The name of a VSAM cluster that you want to reference as was specified in the DEFINE CLUSTER job.

Abbreviation: CL

CLUSTERNAMES
You can use this parameter to list all map names that belong to a specified catalog, sorted by cluster names and then by map names. If no catalog is specified, then all map names for all catalogs will be listed.

Abbreviation: CNAMES

CLUSTERS
You can use this parameter to list all maps, views and columns belonging to all clusters.

Abbreviation: CLS

COLUMN(name)
The name of a column that will be deleted from a map or view, when you specify **VIEW(name)**.

Abbreviation: COL

COLUMNS
You can use this parameter to list all columns belonging to a specific map.

Abbreviation: COLS

DECIMALPOS(number)
You can use this parameter to specify the position of the decimal point for decimal numbers. Positive number for DECIMALPOS indicates the number of digits to the right of the decimal point. Negative number for DECIMALPOS indicates the number of zeros that will be appended to the right of the digits supplied. This parameter can only be applied to the field types: PACKED, UNPACKED, ZONED, and UNZONED.

DECIMALPOS value can only be a decimal number consisting of up to 4 digits including minus sign '-' for negative values. That is, you can specify a number between -999 and 9999. If DECIMALPOS is not specified, it is considered to be zero.

Abbreviation: DECPOS

DESCRIPTION(text)
Description of the field.

Abbreviation: DESC

FIELD(OFFSET(offset),LENGTH(length),TYPE(type))
The VSAM record's offset, length and data type that you want to map. You can write these parameters in any order.

Abbreviation: FLD

LENGTH(length)
Sub-parameter containing the length value for the FIELD parameter.

Abbreviation: L

MAP(name)
The name of a map you want to define or reference.

MAPCOLUMN
A parameter used when you want to define one or more columns for a map.

Abbreviation: MCOL

MAPNAMES
You can use this parameter to list all map names and related cluster names that belong to a specified catalog, sorted by map names within each cluster. If no catalog is specified, then all map names for all catalogs will be listed.

Abbreviation: MNAMES

MAPS
You can use this parameter to list all columns belonging to all maps.

mname
The name of a column you want to define for a map.

name
A string of maximum 44 characters.

NEWNAME(name)
The new name for a column/view/map (which depends upon the previous parameter that was specified).

Abbreviations: NNAME, NEWNM

OFFSET(offset)
Sub-parameter containing the offset value for the FIELD parameter.

Abbreviation: O

POSITION(position)
The position number of a column in a map or view.

Abbreviation: POS

REFCOLUMN(name)
The name of a pointer to the corresponding **MAPCOLUMN**.

Abbreviation: RCOL

TYPE
Sub-parameter containing the data type for the FIELD parameter. The following types are supported: BINARY, STRING, UINTEG, SINTEG, PACKED, UPACKED, ZONED, UZONED.

Note: If you specify the DECIMALPOS parameter, only the following data types can be used: PACKED, UPACKED, ZONED, UZONED

Table 11. Abbreviations for the TYPE Parameter		
Parameter/Value	**Abbreviation**	**Explanation**
TYPE	T	
BINARY	BIN	Binary type
STRING	STR	String type
UINTEG	UINT	Unsigned integer type
SINTEG	SINT	Signed integer type
PACKED	PCKD	Packed type
UPACKD	UPCKD	Unpacked type
ZONED	ZON	Zoned type
UZONED	UZON	Unzoned type

VIEW(name)
The name of a view that you want to define. This parameter requires that you specify **MAP(name)**.

VIEW(name) COLUMNS
You can use this parameter to list all columns belonging to a specific view.

VIEWCOLUMN
The parameter is used to define one or more columns for a view. When defined, each column in a view has a reference to an existing column in a map. The name of a column in VIEWCOLUMN can be different from the name of the referenced column specified in MAPCOLUMN.

Abbreviation: VCOL

VIEWS
You can use this parameter to list all columns belonging to all views.

vname
The name of a column you want to define for a view.

For a discussion of the mapping concept and an example of RECMAP usage, refer to "Mapping VSE/VSAM Data to a Relational Structure" and "Getting Started with the Sample VSAM Applet" in the z/VSE e-business Connectors User's Guide .

REPRO

The REPRO command is used to do any of the following:

- Copy a VSE/VSAM file into another VSE/VSAM file.
- Copy a sequential file on tape or disk into another sequential file on tape or disk.
- Convert a sequential or indexed-sequential file into a VSE/VSAM file.
- Convert a VSE/VSAM or ISAM file into a sequential file.
- Reorganize the records and space in a key-sequenced file into another key-sequenced file. To find out what happens when records from the input file are added to an empty or nonempty output file, see “Reorganizing a File” on page 42.
- Merge two VSE/VSAM files.
- The input file can be on SYSIPT, in which case dname has to be specified as SYSIPT.
- Copy a catalog to a:

  sequential nonVSAM file, or
  VSE/VSAM key-sequenced file, or
  entry-sequenced file,

and copy such files to a catalog. Do not copy the master catalog to a VSE/VSAM file. See “Example 17: Using REPRO to Reload a User Catalog” on page 232.

Note that:

– A catalog should be restored to the same physical volume from which it was copied.

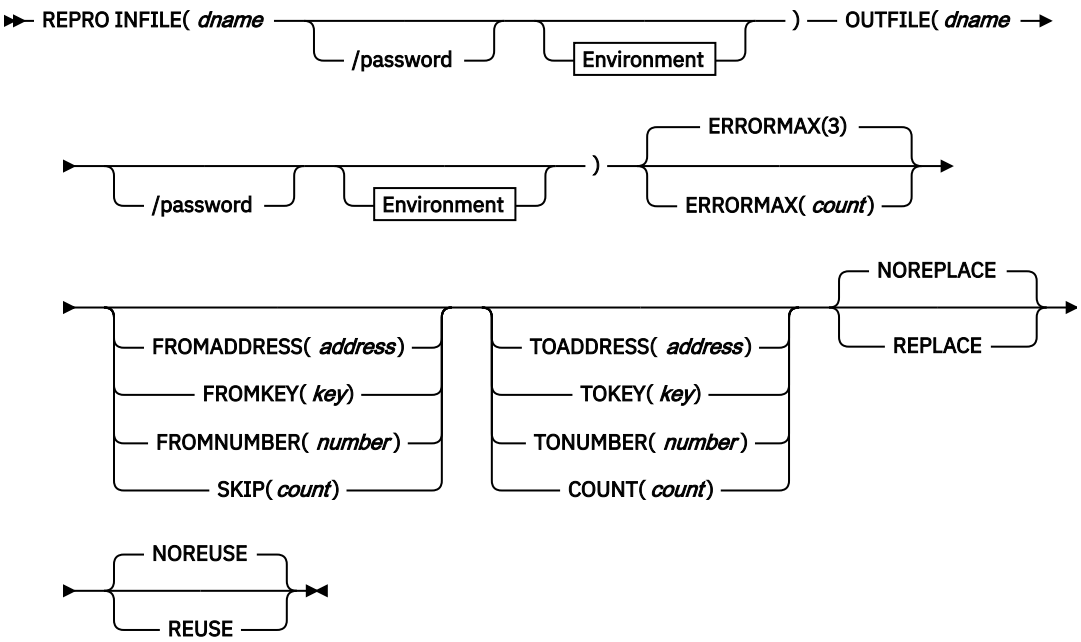
– The previous catalog should *not* be deleted from the disk prior to reloading the catalog.

The original catalog must still exist on the volume prior to the restore. If it does not exist, other VSE/VSAM files on the volume may be inaccessible. For further explanations, refer to “Reloading a Catalog” on page 41.

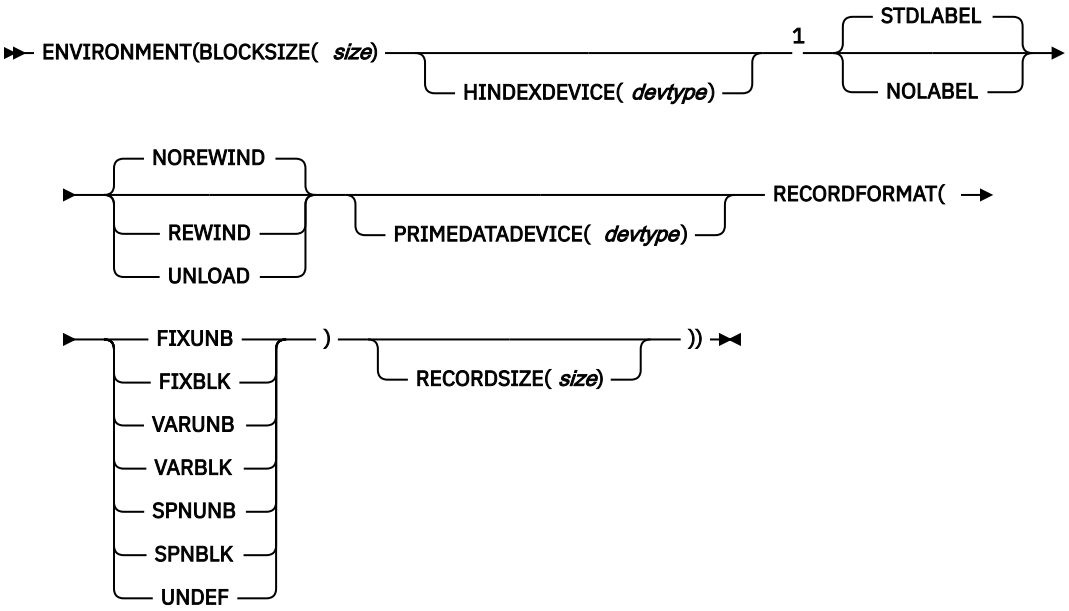
For more information, see also:

- “Loading Records into a File” on page 31
- “Example 5: Loading and Printing of Files” on page 218
- “Example 6: Modifying and Printing the Contents of VSE/VSAM Files” on page 221

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under “PARM” on page 205.) The format of the REPRO command is:



Environment



Notes:

[1] HINDEXDEVICE can be specified only for INFILE.

REPRO Parameters

ENVIRONMENT(subparameters)

describes whether the input or output file is a VSAM or nonVSAM file. If the environment parameter is present, or the INFILE is SYSIPT, then IDCAMS uses a DTF control block to access the file. Otherwise VSAM access (ACB) is used.

When you are copying from a nonVSAM file or a SAM ESDS file and the record format, record size, or block size has been changed using DTF access, then this parameter with the BLOCKSIZE and RECORDFORMAT subparameters is required.

Abbreviation: ENV

BLOCKSIZE(size)

specifies the block size for a nonVSAM file. This parameter is required. The block size you specify depends on the record format of the file:

RECORDFORMAT	BLOCKSIZE
FIXUNB	RECORDSIZE
FIXBLK	RECORDSIZE x no. of records per block
VARUNB[1]	Maximum record size + 8
VARBLK[1]	At least maximum record size + 8
SPNUNB	Full track size depends on a particular CKD device type.
SPNBLK	Full track size depends on a particular CKD device type.

Note:

1. If the output file (OUTFILE) is a VSE/VSAM file or a fixed- or undefined-record-format nonVSAM file, IDCAMS prefixes every logical record with a 4-byte record-length field (RL) that you must account for when you specify BLOCKSIZE (see Figure 8 on page 184).

   In addition, you must add 4 bytes for the block-length field (BL).

For FIXBLK files, substitute the logical record length times the number of records per block. For ISAM FIXUNB files, substitute the logical record length plus the key length. If you are reloading a backup copy of a VSE/VSAM catalog, the block size must be a multiple of 516 plus 4.

The maximum block size is:

1. 32,767 bytes per block for tape file and 32,768 bytes per block for sequential FBA disk files.
2. At least 32,768 bytes per block for sequential CKD disk files up to full track size, but not more than 65,535 bytes per block (64K - 1). Full track size depends on a particular CKD device type.

The minimum block size is 18 bytes.

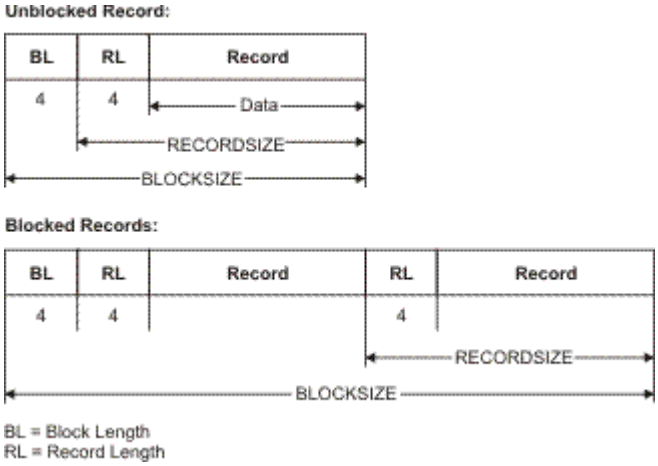


Figure 8. Format of Blocked and Unblocked Records

Abbreviation: BLKSZ

HINDEXDEVICE(devtype)
specifies, for an ISAM input file, the device type of the volume on which the highest index resides.

The possible device types that apply to an ISAM file are: 2314, 2319, 3330, and 3340.

Abbreviation: HDEV

Default: The device type of the highest index defaults to the device type of the volume on which the prime data records reside (see PRIMEDATADEVICE).

NOLABEL | STDLABEL
specifies the type of tape (unlabeled or EBCDIC standard-labeled) to be processed if PRIMEDATADEVICE specifies 2400.

NOLABEL
indicates that an unlabeled tape is to be processed.

For an input file, consider the following: Because the first record of an unlabeled tape may or may not be preceded by a tape mark (user's prerogative), you must take care to properly position the tape to the file you want to process. If a tape mark precedes the file, position the tape either immediately past or immediately before the tape mark. If no tape mark is present, position the tape immediately before the first data record. If the unlabeled tape was created by VSE/VSAM IDCAMS, there is no tape mark preceding the first record of the file.

For an output file, consider the following: You must properly position the tape to the file you want to process. For an output file, IDCAMS does not write a tape mark as the first record. Instead, a data record is written immediately, wherever the tape is positioned. If an output file is to begin somewhere in the middle of the tape, it is your responsibility to position the tape immediately past the ending tape mark of the preceding file. Use the MTC job control statement or command (described in z/VSE System Control Statements) to properly position the tape or to write a tape mark.

If *VSE/Access Control-Logging and Reporting* (IBM VSE/ACLR) is installed, REWIND must be specified for unlabeled tapes.

Abbreviation: NLBL

Restriction:
Although REPRO can create multivolume unlabeled output files, other AMS commands (including REPRO INFILE) can only process single volume unlabeled files.

STDLABEL
indicates EBCDIC standard-labeled tapes are to be processed.

Abbreviation: SLBL

Default: STDLABEL

Restriction: IDCAMS does not support ASCII files or nonstandard labels.

NOREWIND | **REWIND** | **UNLOAD**
specifies the tape positioning action for an OPEN, CLOSE, and EOV (end-of-volume) condition if PRIMEDATADEVICE specifies 2400.

Note: When you use REWIND or UNLOAD (especially with multifile volumes), any OPEN causes the tape to be positioned at load point. You must be sure that the file you want to process is really the first one (at load point) on the tape. Otherwise, specify the NOREWIND option and use the MTC job control statement (described in z/VSE System Control Statements), for example, to properly position the tape.

NOREWIND
specifies that rewind is never to be performed on OPEN, CLOSE, and EOV.

End-of-file considerations: For EBCDIC standard-labeled tapes, the tape is positioned between the two trailing tape marks. For unlabeled tapes, the tape is positioned following the single trailing tape mark.

Abbreviation: NREW

Default: NOREWIND

REWIND
specifies that tapes are to be rewound to load point but not unloaded on an OPEN, CLOSE, or EOV condition.

Abbreviation: REW

UNLOAD
specifies that tapes are to be rewound on an OPEN, and rewound and unloaded on an EOV and CLOSE condition.

Abbreviation: UNLD

PRIMEDATADEVICE(devtype)
specifies the device type of the volume on which the prime data records reside. The possible device types that apply to an ISAM file are 2314, 2319, 3330, and 3340.

You do not have to specify this parameter for a sequential file, unless the file resides on tape (in which case you specify 2400). The 2400 identifies all tape devices that DTFMT supports. You must specify SYS004 in the ASSGN statement for a tape input file; and you must specify SYS005 in the ASSGN statement for a tape output file. For more information on tape, see “Tape Considerations for IDCAMS” on page 12.

Abbreviation: PDEV

Default: 2314

RECORDFORMAT(format)
specifies the format of the records in a nonVSAM file. This parameter is required. For format, substitute one of the following values:

Value		Meaning
FIXUNB	F	Fixed, unblocked
FIXBLK	FB	Fixed, blocked
VARUNB	V	Variable, unblocked
VARBLK	VB	Variable, blocked
SPNUNB	S	Variable spanned, unblocked
SPNBLK	SB	Variable spanned, blocked
UNDEF	U	Undefined

See the BLOCKSIZE and RECORDSIZE subparameters; their values are related to the RECORDFORMAT parameter.

VARBLK must be specified if you are reloading a backup copy of a VSE/VSAM catalog from a nonVSAM file, or if a catalog is to be unloaded into a sequential file.

Abbreviation: RECFM

RECORDSIZE(size)
specifies the length of a logical record for FIXBLK or FIXUNB nonVSAM files. For SPNBLK and SPNUNB nonVSAM files, this parameter is the maximum logical record size plus 4. For UNDEF nonVSAM files, this parameter is the maximum record size. This parameter is not necessary for VARUNB. If you are reloading a backup copy of a VSE/VSAM catalog, or if a catalog is to be unloaded into a sequential file, record size must be 516.

Abbreviation: RECSZ

Default: If RECORDSIZE is not specified, blocking is one record per block. Therefore, RECORDSIZE is equal to block size for files that are FIXUNB, FIXBLK, or UNDEF. RECORDSIZE is equal to block size minus four for files that are VARUNB, VARBLK, SPNUNB, or SPNBLK (see Figure 8 on page 184).

ERRORMAX(count)
specifies the number of so-called "acceptable" errors. Processing is terminated when the number of such errors within the REPRO job step exceeds the value supplied by ERRORMAX. Examples of acceptable errors are:

- Duplicate key (KSDS) or record (RRDS)
- Record out of sequence
- Invalid record length
- Duplicate record in upgrade set

The count can be expressed in decimal (n), hexadecimal (X'n'), or binary form and must not exceed 4 bytes.

Abbreviation: EMAX

Default: 3

FROMADDRESS(address) | **FROMKEY(key)** | **FROMNUMBER(number)** | **SKIP(count)**
specifies the first record to copy from the input file. For the files and keywords allowed in combination with these parameters, see Table 12 on page 186.

Table 12. Files and Keywords with FROM-ADDRESS, -KEY, -NUMBER

Parameter	Used with files	To be used with
FROMADDRESS	KSDS and ESDS	TOADDRESS/COUNT
FROMKEY	KSDS and ISAM	TOKEY/COUNT

Table 12. Files and Keywords with FROM-ADDRESS, -KEY, -NUMBER (continued)		
Parameter	**Used with files**	**To be used with**
FROMNUMBER	RRDS and VRDS	TONUMBER/COUNT
SKIP	ISAM,SAM,VSAM	Any

FROMADDRESS(address)
specifies the RBA of the first record. It must be the beginning of a logical record. If you specify this parameter for a key-sequenced file, the records will be copied in physical sequential order instead of in key order. The address can be expressed in decimal (n), hexadecimal (X‘n'), or binary (B‘n') form.

FROMKEY(key)
specifies the key of the first record. It may be a full key or a generic key—that is, the high-order portion of a key. If you specify a generic key, copying begins at the first record whose key matches the portion of the key you specified. (You must not specify a key longer than that defined for the file. If you do, the file is not copied.) If the specified key is not found, listing begins with the next higher key. The key must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a key must be coded as two single quotation marks if the key is enclosed in single quotation marks; it may contain up to 255 bytes, that is, 510 hexadecimal or 255 EBCDIC characters. If it is specified in hexadecimal, it must be preceded by X and be enclosed in single quotation marks.

FROMNUMBER(number)
specifies the relative-record number of the first record.

SKIP(count)
specifies the number of records you want to skip before beginning to copy records. For example, if you want copying to begin with the 500th record, you specify SKIP(499). The count can be expressed in decimal (n), hexadecimal (X‘n'), or binary (B‘n') form and be no longer than one fullword (4 bytes or 32 bits).

Abbreviations: FKEY, FNUM, and FADDR

Default: If nothing is specified, the copying begins with the first record.

Restrictions: Cannot be used when input file is a catalog.

INFILE(dname)
describes the input file. If the file is defined in a user catalog which is not the job catalog, you must explicitly name the user catalog by specifying the CAT parameter in the file's DLBL job control statement. (The user catalog's DLBL statement does not require an associated EXTENT statement.) When using REPRO to load an ISAM file to be run under the ISAM Interface Program (IIP), the INFILE must be in ISAM format. Do not use REPRO to copy the ISAM file onto tape before loading it because copying it onto tape makes it a SAM file.

dname - specifies the file name of the DLBL (an associated EXTENT statement is required only if the input file is a nonVSAM file) or TLBL job control statement that identifies the file to be copied. The input file can be on SYSIPT, in which case dname has to be specified as SYSIPT. You can process a base cluster in alternate key sequence by specifying a path name as file ID the DLBL statement. This is a required parameter.

If an unlabeled tape is used (NOLABEL option), a TLBL statement is not required. However, because dname is a required positional parameter, a dummy file name of your choosing must be specified.

password - the read or higher-level password of the password-protected file to be copied. If a catalog is to be copied, the catalog's master password is required. For a path, it is the read or higher-level password of the path.

Abbreviation: IFILE

OUTFILE(dname)
dname - specifies the file name of the DLBL (an associated EXTENT statement is required only if the output file is a sequential file) or TLBL job control statement that identifies the output file. ISAM files cannot be specified as output files. This is a required parameter. For VSE/VSAM files, the file ID may be that of a path. A master catalog must be copied to a sequential file.

If an unlabeled tape is used (NOLABEL option), a TLBL statement is not required. However, because dname is a required positional parameter, a dummy (fictitious) file name of your choosing must be specified.

If the file is defined in a user catalog which is not the job catalog, you must explicitly name the user catalog by specifying the CAT parameter in the file's DLBL job control statement. (The user catalog's DLBL statement does not require an associated EXTENT statement.)

password - the update or higher-level password of the VSE/VSAM file used for output if it is password-protected. If a catalog is to be reloaded, the catalog's master password is required; for a path, it is the update or higher-level password of the path.

Abbreviation: OFILE

REPLACE | **NOREPLACE**
specifies whether an input record which has a key (key-sequenced file) or record number (relative-record file) identical to the key or record number of a record in the output file is to replace that record.

If you specify NOREPLACE, VSE/VSAM issues a warning message that identifies the key or record number of the duplicate record; processing continues with the next record. Do not specify REPLACE if the output file is:

- An entry-sequenced or sequential file
- A path over an alternate index
- A base cluster with a unique-key alternate index in the upgrade set
- An empty file to be loaded

Abbreviations: REP and NREP

Default: NOREPLACE

REUSE | **NOREUSE**
specifies whether the output file will be opened with the reset option (that is, with its “high-used RBA” set to 0). If REUSE is specified and the file is not empty and is defined as nonreusable (NOREUSE), REPRO terminates. If DISP=OLD is specified in the DLBL statement, REUSE does not reset the high-used RBA because the DLBL specification overrides the REPRO command.

Abbreviations: RUS and NRUS

Default: NOREUSE

TOADDRESS(address) | **TOKEY(key)** | **TONUMBER(number)** | **COUNT(count)**
specifies the last record to copy in the file. The last record must follow the first record. For the files and keywords allowed in combination with these parameters, see Table 13 on page 188.

Table 13. Files and Keywords with TO-ADDRESS, -KEY, -NUMBER, COUNT

Parameter	Can be used with	To be used with
COUNT	ISAM,SAM,VSAM	Any
TOADDRESS	KSDS and ESDS	FROMADDRESS/SKIP
TOKEY	KSDS and ISAM	FROMKEY/SKIP
TONUMBER	RRDS and VRDS	FROMNUMBER/SKIP

TOADDRESS(address)
specifies the RBA of the last record. Unlike FROMADDRESS, the RBA does not need to be the beginning of a logical record. The entire record containing the specified RBA is printed. If you specify this parameter for a key-sequenced file, the copying will be in physical sequential order instead of in key order. The address can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

TOKEY(key)
specifies the key of the last record. It may be a full key or a generic key - that is, the high-order portion of a key. If you specify a generic key, copying stops after the last record is copied whose key matches the portion of the key you specified. (You must not specify a key longer than that defined for the file. If you do, the copying is not performed.) If the specified key is not found, copying ends with the next lower key. The key must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark within a key must be coded as two single quotation marks if the key is enclosed in single quotation marks; the key may contain up to 255 bytes, that is, 510 hexadecimal or 255 EBCDIC characters. If it is specified in hexadecimal, it must be preceded by X and be enclosed in single quotation marks.

TONUMBER(number)
specifies the relative-record number of the last record.

COUNT(count)
specifies the number of records to be copied. The count can be expressed in decimal (n), hexadecimal (X'n'), or binary form and be no longer than one fullword (4 bytes or 32 bits).

Abbreviations: TADDR and TNUM

Default: If nothing is specified, the copying ends with the last record.

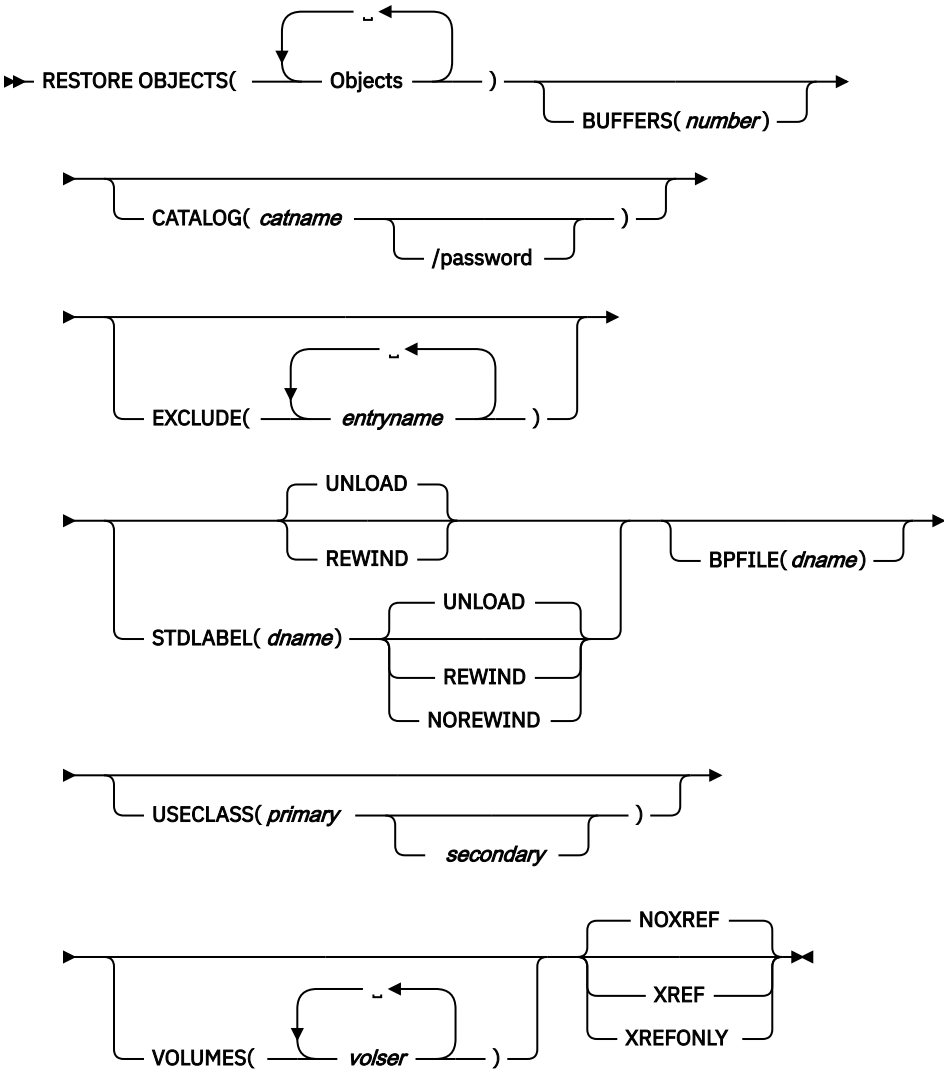
Restrictions: Cannot be used when input file is a catalog.

RESTORE

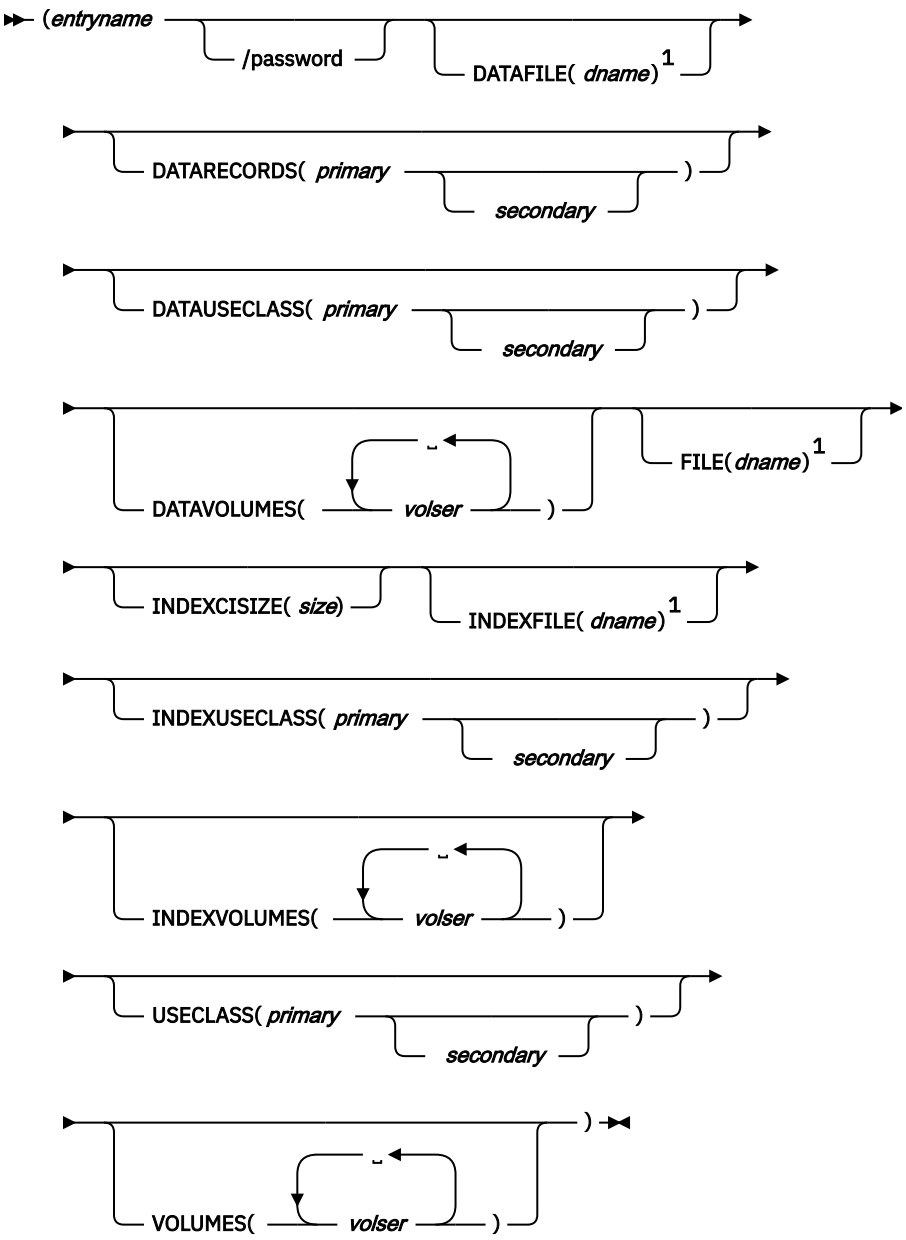
The RESTORE command (abbreviated form: RST) is used to reinstall the objects of a backup file that was produced through the BACKUP command. All objects or individual objects of the backup file can be restored. Generic restoration is also possible.

All alternate indexes contained on the backup file are automatically restored with their related base cluster. Also, all paths are restored when their associated alternate index or path entry cluster is restored.

The format of the RESTORE command is:


RESTORE OBJECTS(
Objects
)
BUFFERS(number)
CATALOG(catname
/password
)
EXCLUDE(
entryname
)
UNLOAD
REWIND
BPFILE(dname)
STDLABEL(dname)
UNLOAD
REWIND
NOREWIND
USECLASS(primary
secondary
)
VOLUMES(
volser
)
NOXREF
XREF
XREFONLY

Objects



Notes:

[1] DATAFILE, FILE, and INDEXFILE are required under certain conditions.

For more information and examples, see "Using BACKUP and RESTORE" on page 48 and "Examples of Functions for RESTORE" on page 253.

RESTORE Parameters: Summary

File Modification at Restoration: The following parameters are required only if you want to change the characteristics of the specified object when it is redefined in the catalog during restoration:

VOLUMES
DATAVOLUMES
INDEXVOLUMES
USECLASS
DATAUSECLASS
INDEXUSECLASS
DATARECORDS

INDEXCISIZE

By means of these parameters, either the whole object or its data or index component can be moved onto different volumes or to different space classes.

Performance Degradation: If you specify that a file is to reside on a volume of a different device type, or if you specify DATARECORDS or INDEXCISIZE, VSE/VSAM usually reorganizes the file's CAs or CIs. You will experience degraded performance compared to running RESTORE without such modifications.

Changing a file or component's use class has no noticeable effect on performance.

Global/Local Specification: Note that the VOLUMES and the USECLASS parameters can appear on two levels: they can be effective only for the entryname for which they are specified (local specification), or they can be effective for all specified entrynames unless specifically overridden (global specification). For example:

```
RESTORE OBJECTS((A VOLUMES(VOL001))
                  .
                  .
                  .
                  .
                )
                VOLUMES(VOL002)
```

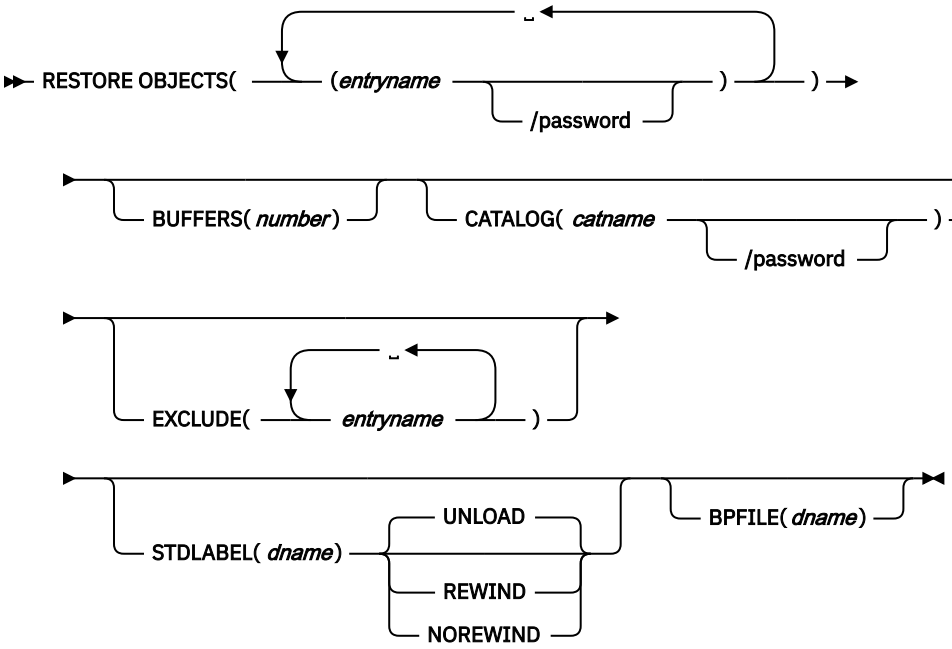
In this example, all objects will be restored to VOL002, except object A which will be restored onto VOL001.

Unique Files: The following parameters are only applicable to a unique file:

FILE
DATAFILE
INDEXFILE

These parameters are used to identify the DLBL/EXTENT statements that contain the space information for a unique file or its unique components. If no unique files are to be restored, you can ignore these parameters.

If you do not have to specify any of the parameters listed above, the RESTORE command is reduced to the following format:



The following table shows which keywords can be used with the RESTORE command. The "Use" column specifies whether the parameter can apply to the object as a whole (O), to the data component (D), or to the index component (I).

RESTORE Parameter	Abbr.	Use	Example	Notes
Required Parameters:				
OBJECTS	OBJ		OBJ((...))	
FILE		ODI	FILE(FILA)	1.
DATAFILE	DFILE	D	DFILE(DAT)	1.
INDEXFILE	IXFILE	I	IXFILE(IND)	1.
Optional Parameters:				
BPFILE			BPFILE(BACKIN)	
BUFFERS	BFRS		BFRS(2)	
CATALOG	CAT		CAT(UCT/PWD)	
DATARECORDS	DREC	D	DREC(500 100)	
DATAUSECLASS	DUSCL	D	DUSCL(7 P)	2.
DATAVOLUMES	DVOL	D	DVOL(DVOL)	2.
EXCLUDE	EXCL	O	EXCL(FLE.A)	
INDEXCISIZE	IXCISZ	I	IXCISZ(10 24)	
INDEXUSECLASS	IXUSCL	I	IXUSCL(7 P)	2.
INDEXVOLUMES	IXVOL	I	IXVOL(IVOL)	2.
NOXREF	NXREF		NXREF	
STDLABEL	SLBL		SLBL(TAPE)	
USECLASS	USCL	ODI	USCL(7 P)	2.
VOLUMES	VOL	ODI	VOL(VOL1)	2.
XREF			XREF	

RESTORE Parameter	Abbr.	Use	Example	Notes
XREFONLY	XREFY		XREFY	

Notes:

1. Required only if a unique file is to be restored.
2. Required only if the object or its data or index component is to be moved to a different space class or volume.

RESTORE Parameters

OBJECTS(entryname)
names the object(s) to be restored.

entryname - is the name of an object or set of objects to be restored. It can be either the name of a single object or a generic name.

A generic name is an entryname that contains an asterisk as the last name segment. It represents all entrynames contained on the backup file that start with the name segment(s) preceding the * in the generic name. Specify one or more complete name segments. Be sure to include the final period before the asterisk.

Note that a generic name is an abbreviation for a set of *explicitly* named objects.

If only one entry name is specified, the parentheses surrounding the entryname/password may be omitted.

If you want to restore all objects of the backup file, specify * for entryname.

Up to 255 entrynames can be specified with a single command.

password - specifies the password of a password-protected object contained in the catalog that is to be replaced by an object of the same name contained on the backup file. The password must be either the master password of the object to be replaced or the catalog's master password.

Thus, an object that was backed up without its passwords (because only the read password was specified) can only replace the old copy if the master password of the password-protected old copy or the catalog's master password is specified.

If a generic name is used, either the master password of all objects to be replaced must be the same, or the master password of the receiving catalog must be specified.

Abbreviation: OBJ

BPFILE(dname)
indicates that the backup file resides on disk.

dname specifies the *filename* of the DLBL statement that contains the necessary label and extent information. If the BPFILE parameter is not specified, and SYS004 is assigned to a disk device, a dname of BACKIN is assumed.

BUFFERS(number)
specifies the number of common data buffers to be used for restoration. The size of the buffers is the same as that used for the associated backup operation.

For a backup file *on tapes*, the number of buffers specified should not exceed the number of buffers specified in the corresponding BACKUP command; if it does, the value is automatically reduced to that value.

For a backup file *on disks*, any number of buffers greater than 3 can be specified. However, specifying a high number of buffers does not increase the performance of the restoration process, because read operations for a disk resident backup file can only be performed one after the other (that is, read operations cannot overlap one another).

If BUFFERS parameter is not specified, the number of buffers for the corresponding backup operation is used for both tape resident backup files and disk resident backup files.

Abbreviation: BFRS

CATALOG(catname)
specifies the name and the password of the catalog in which the restored objects are to be defined. You must specify this parameter if the catalog that is intended to receive the VSE/VSAM objects to be restored is not the master catalog or the job catalog. This parameter must also be specified if the receiving catalog is password protected, or you will be prompted for the password. The password must either be the update password or a higher-level password of the catalog.

Abbreviation: CAT

DATAFILE(dname)
is effective only for a file with a UNIQUE data component.

dname - identifies the DLBL/EXTENT statements that contain the space information for the UNIQUE data component.

DATAFILE or FILE must be specified for a UNIQUE data component. If both are specified, DATAFILE takes effect for the data component and FILE is ignored as far as the data component is concerned.

DATAFILE can only be specified for individual VSE/VSAM objects and not on the global level.

Abbreviation: DFILE

DATARECORDS(primary)
specifies the amount of space, in units of records, to be allocated for the data component.

DATARECORDS is used to change the storage allocation for an object.

Specify the amount in decimal (n), hexadecimal (X'n'), or binary (B'n'). VSE/VSAM multiplies the values specified for primary and secondary space by the average record size of the file to determine the allocation amount for the data component. If the file is to contain free space, specify an amount large enough to accommodate the free space as well.

Specifying DATARECORDS overrides both primary and secondary allocation amounts for the data component. If you specify only the primary amount, the secondary amount defaults to zero.

For indexed files whose data and index components are to reside in space of the same use class (specified either at restoration or when the file was originally defined), VSE/VSAM calculates the amount of space to be allocated for the index component. If data and index are to reside in space of different use classes, the space allocated for the index component will be the same as when the object was backed up (that is, index space allocation will be unaffected by the DATARECORDS parameter).

You may specify this parameter for only specific entrynames (local specification). It may not be specified globally to apply to all objects. If you do not specify this parameter, the allocation size(s) saved during backup will be used to define the object for restoration.

Be aware that specifying DATARECORDS usually causes VSE/VSAM to use a new CA size for the data component. When this file modification occurs, you will experience degraded performance compared to RESTORE without file modification.

Abbreviation: DREC

DATAUSECLASS(primary)
can be specified only for individual objects. (It cannot be specified globally to apply to all objects.) It is used to modify the space classes for primary and secondary allocation of the data component of the restored object as follows:

Primary allocation of space:

```
0    The object is to occupy class-0 data space.
     Class-0 is for general use and is the default.
     (Data space defined under DFSMSdfp VSAM is treated as class-0.)
```

```
1-7  The object is to occupy any user-defined space
     class between 1 and 7.
```

Secondary allocation of space:

```
0    Class-0 data space is to be used by the object.
P    The primary space classification is to be used.
     P is the default.
```

The space classes for the index component are not modified. The specification of DATAUSECLASS overrides any global specification or an appropriate local (pertaining to a particular entryname) specification of the USECLASS parameter as far as the data component is concerned.

Abbreviation: DUSCL

DATAVOLUMES(volser)
can be specified only for individual objects. (It cannot be specified globally to apply to all objects.) It is used to move the data component of the restored object to a set of volumes that is different from the volume(s) from which the data component was backed up. The volumes need not be of the same device type as the backup volumes.

The volumes of the index component are not affected by this parameter.

The specification of DATAVOLUMES overrides any global specification or an appropriate local (pertaining to a particular entryname) specification of the VOLUMES parameter as far as the data component is concerned.

volser denotes the list of volume serial numbers of the new volume set for the data component.

Abbreviation: DVOL

EXCLUDE(entryname)
specifies a list of VSE/VSAM objects that are not to be restored, though a generic name in the OBJECTS parameter may include these VSE/VSAM objects.

entryname - is the name of a single object or a generic name. Up to 255 entrynames can be specified.

If an object is excluded from restoration, any alternate indexes and paths defined for the object are also excluded, unless they (the alternate indexes) are explicitly named in the OBJECTS parameter.

Note that a generic entry in the OBJECTS parameter is interpreted as an abbreviation for the set of entrynames expressed by the generic name. All objects represented by a generic name are considered explicitly specified so that an alternate index represented by a generic name is restored even if its base cluster is excluded from restoration.

If an object is restored but an alternate index defined for it is excluded from restoration, the alternate index (and any paths associated with it) are automatically deleted.

Abbreviation: EXCL

FILE(dname)
can be specified only for individual VSE/VSAM objects. (It cannot be specified globally to apply to all objects.)

dname - identifies the DLBL/EXTENT statements that contain the space information for all UNIQUE components of a file to be restored.

If both components of the file are unique and reside on the same volume, you must specify at least two of the three parameters FILE, DATAFILE, or INDEXFILE, identifying different

INDEXFILE parameter DLBL/EXTENT statements for the data and index components. FILE then applies to the components for which no explicit DATAFILE or INDEXFILE parameter is specified.

Abbreviation: none

INDEXCISIZE(size)
specifies, in number of bytes, the index CI size to be used in definition and restoration of an individual object. Specify the value in decimal (n), hexadecimal (X‘n’), or binary (B‘n’). The size can be any

multiple of 512, up to 8192 bytes. If an improper multiple is coded, VSE/VSAM selects the next higher multiple. INDEXCISIZE may be specified for only specific entrynames (local specification). It cannot be specified globally to apply to all objects.

If you do not specify this parameter, the index CI size saved during backup is used to define the object for restoration. If you specify this parameter for a file that is not a KSDS, it is ignored.

You will probably not have to specify this parameter often, if at all. Typically you would use it if you have restored a file to a different device type and discovered (from LISTCAT output) that the restored file has increased unreasonably in size.

Be aware that specifying INDEXCISIZE forces VSE/VSAM to do a CI reorganization. This file modification will cause degraded performance compared to RESTORE without file modification.

Abbreviations: IXCISZ, IXCNVSZ, or INDEXCNVSIZE

INDEXFILE(dname)
is effective only for a file that has a UNIQUE index component.

dname - identifies the DLBL/EXTENT statements that contain the space information for the UNIQUE index component.

INDEXFILE or FILE must be specified for a UNIQUE index component. If both are specified, INDEXFILE takes effect for the index component and FILE is ignored as far as the index component is concerned.

INDEXFILE can only be specified for individual VSE/VSAM objects. (It cannot be specified globally to apply to all objects.)

Abbreviation: IXFILE

INDEXUSECLASS(primary)
can be specified only for individual objects. (It cannot be specified globally to apply to all objects.) It is used to modify the space classes for primary and secondary allocation of the index component of a key-sequenced file or an alternate index.

The space classes for the data component are not modified.

The specification of INDEXUSECLASS overrides any global specification or an appropriate local (pertaining to a particular entryname) specification of the USECLASS parameter as far as the index component is concerned.

Primary and secondary have a similar meaning as described under the DATAUSECLASS parameter.

Abbreviation: IXUSCL

INDEXVOLUMES(volser)
can be specified only for individual objects. (It cannot be specified globally to apply to all objects.) It is used to move the index component of a key-sequenced file or an alternate index to a set of volumes that is different from the volume(s) from which the index component was backed up. The volumes need not be of the same device type as the backup volumes.

The volumes of the data component are not affected by this parameter.

The specification of INDEXVOLUMES overrides any global specification or an appropriate local (pertaining to a particular entryname) specification of the VOLUMES parameter as far as the index component is concerned.

volser denotes the list of volume serial numbers of the new volume set for the index component.

Abbreviation: IXVOL

STDLABEL(dname)
is required if the backup file resides on tapes and if it contains standard (EBCDIC) tape labels. In this case, dname specifies the file name of the TLBL statement that contains the necessary label information. STDLABEL is also required for multiple catalog restores (see NOREWIND).

Abbreviation: SLBL

UNLOAD | REWIND
UNLOAD specifies that backup file tapes are rewound on an OPEN, and rewound and unloaded on an EOV and CLOSE condition.

Abbreviation: UNLD

REWIND specifies that backup file tapes are to be rewound to load point but not unloaded on an OPEN, CLOSE, or EOV condition.

Abbreviation: REW

Default: UNLOAD

NOREWIND
RESTORE keeps the tape positioned afer the step has finished. This is then the start position for the next RESTORE (possibly of another catalog) from the same tape in a subsequent job step. NOREWIND can be used only in conjunction with STDLABEL.

Note: If a restore job is for more than one catalog, the NOREWIND parameter must be used for *all* job steps, including the last. You cannot use REWIND or UNLOAD within such a job stream.

Abbreviation: NREW

USECLASS(primary)
can be specified for either a particular entryname or all entrynames. If specified on a global level, it is effective for all specified objects for which an individual USECLASS is not also specified. When specified for a particular entryname, the USECLASS parameter is effective only for the entry it is associated with and overrides any global specification.

With this parameter, you can modify the space classes for primary and secondary allocation of all components of the explicitly named restored objects. Primary and secondary have the same meaning as described under the DATAUSECLASS parameter.

Abbreviation: USCL

VOLUMES(volser)
can be specified for either a particular entryname or all entrynames. If specified on a global level, it is effective for all specified objects, except:

- Automatically restored objects
- Objects whose global specification is overridden by a specification for the particular entryname.

When specified for a particular entryname, the VOLUMES parameter applies only to the entry it is associated with and overrides any global specification.

With this parameter, you can move all components of the explicitly named restored objects to a set of volumes different from the one from which the objects were backed up. The volumes need not be of the same device type as the backup volumes. The VOLUMES parameter need not be specified if the backup copy is to be restored to the same volser from which the backup was made.

volser denotes the list of volume serial numbers of the new volume set.

Abbreviation: VOL

NOXREF | XREF | XREFONLY
Specifies whether cross-reference listings are to be produced. For the description of what type of listings can be created, refer to “Backup Cross-Reference Listings” on page 51. For examples of listings, refer to “Backup Cross-Reference Listings” on page 240.

NOXREF specifies that cross-reference listings will not be produced but object restoration will be performed.

Abbreviation: NXREF

XREF specifies that both cross-reference listings will be produced and object restoration will be performed.

XREFONLY specifies that only cross-reference listings will be produced but no object restoration will be performed.

Abbreviation: XREFY

Default: NOXREF

SNAP

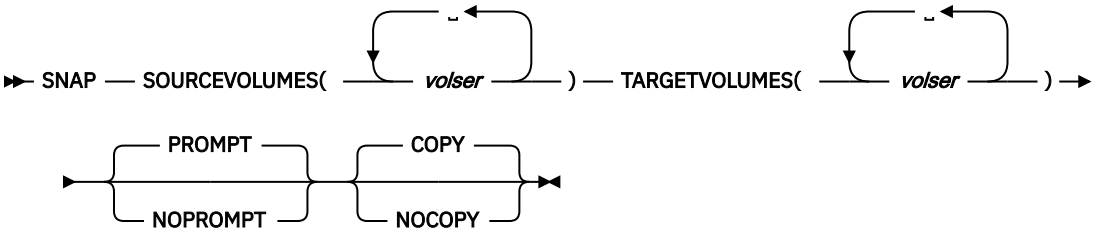
The IDCAMS SNAP command is based on the IXFP SNAP function and thus can be used for full pack volumes on the IBM TotalStorage™ Enterprise Storage Servers (ESS). The SNAP command produces a snapshot of specified source volumes and, if issued with the COPY parameter, copies the snapshot onto target volumes. For a general overview of the command, refer to "Performing an IDCAMS SNAP (FlashCopy)" in VSE/VSAM User's Guide and Application Programming.

If SNAP is issued with the COPY parameter, copying is performed in the background and can take up to tens of minutes for large disks. When copying is completed, the ESS resources used to support the flashcopy are freed. You can stop the process of copying before its completion. This can be done by using the IDCAMS SNAP DDSR command. Note that in this case the content of the target volumes is not predictable and the target volumes should be reinitialized.

If SNAP is issued with the NOCOPY parameter, copying is not performed. When the snapshot is no longer needed, you must delete it. This is required to free the ESS resources used by the snapshot and can be done by the IDCAMS SNAP DDSR command. After this, the target volumes have to be re-initialized because they might be used for keeping a copy of the source data in case of ESS cache shortage and their contents is not predictable.

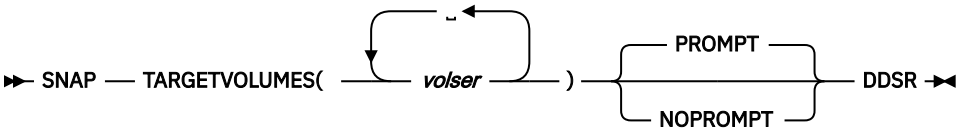
SNAP implicitly uses ESS global resources and is therefore controlled by the system administrator. The administrator can restrict access to SNAP functionality as described in "Controlling Access to the IDCAMS SNAP Command" in VSE/VSAM User's Guide and Application Programming.

The format of the SNAP command for creating a snapshot is the following:



SNAP Parameters

The format of the SNAP command for terminating the FlashCopy relation is the following:



SOURCEVOLUMES(volser[volser...])
is a list indicating the volumes from which the snapshot is to be taken.

Abbreviation: SVOLUME or SVOL

TARGETVOLUMES(volser[volser...])
is a list indicating the volumes to which the snapshot is to be written.

Abbreviation: TVOLUME or TVOL

NOPROMPT | PROMPT
NOPROMPT prevents decision-type messages from being issued, while PROMPT permits them.

Default: PROMPT

COPY | NOCOPY
COPY specifies that the snapshot of source volumes created by the SNAP command is to be copied onto target volumes. NOCOPY specifies that the snapshot of source volumes is not copied to target volumes.

Default: COPY

DDSR
SNAP DDSR (Delete Data Space Request) deletes the snapshot created by a previous SNAP COPY/NOCOPY command and frees the ESS resources used by it. After using the SNAP DDSR command, the content of target volumes is not predictable, you have to reinitialize these volumes before using them in another job. For a snapshot created with the SNAP NOCOPY command, it is mandatory to delete it by SNAP DDSR. Otherwise, the ESS resources will never be freed and this will affect the ESS performance.

DROP can also be used as an alias of the DDSR parameter.

VERIFY

The VERIFY command (VFY) is used to compare the end-of-file information as it is stored in a VSE/VSAM catalog with the end-of-file indicator(s) in the file itself. If the information in the catalog does not agree with the end-of-file indicator(s) in the file, the catalog information is corrected.

The VERIFY command is commonly used to correct the catalog description of a file that was not properly closed (because of either a system or user error). For most files, VSE/VSAM automatically invokes VERIFY the next time it opens the file. However, VSE/VSAM cannot do automatic verification if:

1. The file is an entry-sequenced file being opened for control-interval access (ACB MACRF=CNV).
2. The file is a SAM ESDS defined in non-CI format (RECORDFORMAT=NOCIFORMAT). For a CI-format SAM ESDS, the EOF indicator in the catalog is not updated. This is because the file is always loaded and extended in SPEED mode.
3. The file was defined in SPEED mode, and initial load failed.

In case 1, you should issue VERIFY to validate catalog records if you know that the data in the file is in VSE/VSAM control-interval format. In case 2, VERIFY is not useful. In case 3, you must reload the file.

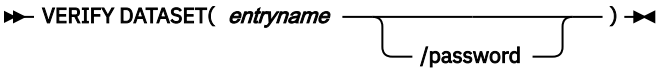
Clusters, alternate indexes, their components, paths defined over base clusters, and catalogs can be verified; paths defined over an alternate index cannot be verified. To avoid possible errors, do not specify the file (key-sequenced or alternate index) to be verified at the data component or index component level; always specify it at the cluster level.

VERIFY does not correct statistics that may be wrong as a result of a system failure. If the VERIFY command is to be used to recover from a failure during initial loading of a file, the RECOVERY parameter (see DEFINE CLUSTER) must have been specified or defaulted to when the file was defined.

If the SPEED parameter was specified when the file was defined, and VERIFY is executed after a failure during initial load, no attempt is made to compare and recover the end-of-file indicators. No message is issued to describe this condition (SPEED mode and no end-of-file recovery). If the file is being extended (for example, when loading is resumed or records are added), VSE/VSAM always extends in RECOVERY mode. Thus, if the file is not properly closed, VSE/VSAM performs automatic verification the next time the file is opened.

Because VERIFY opens the file for output, the password is required. Also, for a reusable file, VSE/VSAM uses the default dispositions (NEW,KEEP). If you have a NOALLOCATION file previously closed with DISP=(,DELETE), VERIFY allocates space to the file and then keeps the file.

If you want to verify the syntax of this command before modifying data, run a trial job specifying PARM SYNCHK. (For further information, see the description of the SYNCHK parameter under "PARM" on page 205.) The format of the VERIFY command is:



DATASET(entryname)
entryname - specifies the entryname of the object to be verified.

The object to be verified must be in (or be) the default catalog. (The default catalog is (1) the job catalog if one is used, or (2) the master catalog.)

password - is the control or master password of a password-protected cluster, alternate index, or component, or the master password of a password-protected catalog.

Abbreviation: DS

Modal Commands and their Formats

The modal commands discussed in the following are:

IF-THEN-ELSE command sequence
To control functional command execution by testing condition codes.

SET command
To set or reset condition codes.

PARM command
To specify options for diagnosis tools, syntax checking, printed output, and input record margins.

IF-THEN-ELSE

Condition Codes

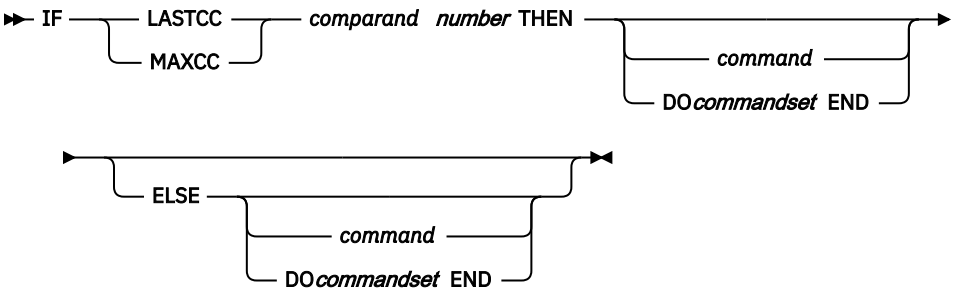
The condition codes that are tested in the IF-THEN-ELSE command sequence are:

- 0, which indicates that the function was executed as directed and expected. Informational messages may have been issued.
- 4, which indicates that a problem was met in executing the function, but it was possible to continue and complete the function. The results might not be exactly what you wanted, but no permanent harm has been done. For example, with LISTCAT, the system might have been unable to locate an entry to be listed. A warning message was issued.
- 8, which indicates that a requested function was completed, but major specifics were unavoidably bypassed. For example, an entry to be deleted or altered could not be found in the catalog, or a duplicate name was found while an entry was being defined.
- 12, which indicates that the requested function could not be performed. This condition code might be set because IDCAMS could not open a user file in REPRO or PRINT because you specified FROMKEY or TOKEY when the file was neither ISAM nor KSDS, or because you named an unavailable catalog in the CATALOG parameter of DEFINE, ALTER, etc.
- 16, which indicates that a severe error occurred and the remainder of the command stream was flushed. This condition code might be set because a system output file could not be opened, an unrecoverable error occurred in a system file, or IDCAMS encountered improper command sequences.

IF-THEN-ELSE Command Sequence

The IF-THEN-ELSE command sequence is used to control command execution.

The format of the IF-THEN-ELSE command sequence is:



where:

IF
specifies that one or more functional commands are to be executed based on a test of a condition code. The condition code is set by a SET command or is set by IDCAMS to reflect the completion status of a previous functional command.

LASTCC
specifies that the condition code that resulted from the immediately preceding function command is to be compared as indicated by *comparand number* to determine whether the THEN action is to be performed. LASTCC is initialized to zero upon entry to IDCAMS. See "IF-THEN-ELSE" on page 201 for the meaning of condition codes.

MAXCC
specifies that the maximum condition code that resulted from previous functional commands or SET commands is to be compared as indicated by comparand number to determine whether the THEN action is to be performed. MAXCC is initialized to zero upon entry to IDCAMS. See "IF-THEN-ELSE" on page 201 for the meaning of condition codes.

comparand - specifies the comparison to be made between the condition code specified by LASTCC or MAXCC and the following number. This can be any of six possible comparisons:

Equal, specified as "=" or "EQ"
Not equal, specified as "NE"
Greater than, specified as ">" or "GT"
Less than, specified as "<" or "LT"
Greater than or equal, specified as "GE"
Less than or equal, specified as "LE"

number - specifies the decimal integer to which LASTCC or MAXCC is to be compared. The number can be up to ten digits long. Values greater than 16 are reduced to 16. See "IF-THEN-ELSE" on page 201 for the meaning of condition codes.

THEN
specifies that a single command or a group of commands (introduced by DO) are to be executed if the IF statement is true. THEN can be followed by another IF statement.

DO-END
(see "DO-END Command Sequence" on page 204).

ELSE
specifies that a single command or a group of commands (introduced by DO) are to be executed if the IF statement is false. ELSE can be followed by another IF statement. See "Nested IF Statements" below for a discussion of IF following THEN or ELSE.

If you want to copy the file identified by MYDATA01 only if the last condition code was zero, specify:

```
IF LASTCC = 0 -
THEN REPRO INFILE(MYDATA01) OUTFILE(MYOUT02)
```

Null Clauses

A null clause is a THEN or ELSE clause that is not followed by a command or a continuation character. If no THEN action is required, the null THEN must be specified; a null ELSE need be specified only if required to balance THENs and ELSEs in nested IF statements.

If you want to indicate a null ELSE clause, specify:

```
ELSE
```

A null ELSE clause indicates that no action is to be taken if the IF statement is false.

If you want to indicate a null THEN clause, specify:

```
IF...THEN
ELSE...
```

A null THEN clause indicates that no action is to be taken if the IF statement is true.

This example illustrates a null ELSE clause.

```
IF LASTCC = 4 THEN PRINT ...
ELSE
```

No action is taken if LASTCC does not equal 4.

The next example illustrates a null THEN clause.

```
IF LASTCC = 4 THEN
ELSE PRINT...
```

No action is taken if LASTCC does equal 4.

The examples above illustrate THEN and ELSE clauses that have been made null simply by the absence of a following command or continuing hyphen. Alternatively, a semicolon may be used (because a semicolon also terminates a command).

Nested IF Statements

When an IF statement appears in a THEN or ELSE clause, it is called a nested IF statement. Nested IF statements are coded exactly like unnested IF statements. The maximum number of IF statements in a nest is 10, including the unnested IF.

Within a nest of IF statements, every ELSE is matched with the nearest preceding unmatched THEN. If an IF statement does not require an ELSE clause, code a null ELSE clause after it, unless no other ELSE clause follows. If an ELSE clause follows, it would be taken for a missing preceding one. The following example below has a null ELSE clause to avoid this:

```
IF LASTCC > 4        -
  THEN IF MAXCC < 12 -
    THEN REPRO ...
    ELSE DELETE ...
  ELSE IF LASTCC = 4 -
    THEN
    ELSE PRINT ...
```

The example above specifies that if LASTCC is greater than 4, then MAXCC is to be tested. If MAXCC is less than 12, the REPRO command is executed, while if MAXCC is 12 or greater, the DELETE command is executed instead. However, if LASTCC is 4 or less, LASTCC is tested for being exactly 4. No action is to be taken in this case. If, however, LASTCC is less than 4, the PRINT command is to be executed.

The following example is similar to the first, except for the use of a null ELSE clause:

```
IF LASTCC > 4        -
  THEN IF MAXCC < 12 -
    THEN REPRO ...
    ELSE
```

```
    ELSE IF LASTCC = 4 -
      THEN PRINT ...
```

If LASTCC is greater than 4, but MAXCC is 12 or greater, no functional command is executed. The null ELSE clause is employed here to cause the next ELSE to correspond to the first THEN.

DO-END Command Sequence

DO
specifies that the group of commands that follow is to be treated as a single unit (to be executed as a result of a single test in an IF command). The set of commands is terminated by END. The first command following a DO must begin on a new line.

END
specifies the end of a set of commands initiated by the nearest preceding unended DO. END must be on a line by itself.

If THEN or ELSE is not followed by DO, by a continuation character, or by a command in the same record, the THEN or ELSE is null. It results in no action.

If you want to list a catalog and print a file if the maximum condition code is zero, specify:

```
IF   MAXCC = 0  THEN  DO
  LISTCAT  CATALOG (AMASTCAT/MST27) ENT (MN01.0005)
  PRINT INFILE (AJK006)
  END
ELSE ....
```

If you want to list a catalog and print a file if the last condition code is zero, but otherwise list its catalog entry before and after a VERIFY command, specify:

```
IF  LASTCC = 0 THEN   DO
    LISTCAT ...
    PRINT INFILE (AJK006)
    END
  ELSE DO
    LISTCAT ENTRY (AJK006) ALL
    VERIFY DATASET (FILE001)
    LISTCAT ENTRY (AJK006) ALL
               END
```

DO groups and nested IF statements may be combined. In the next example, a file is printed if LASTCC is equal to zero and is reproduced if MAXCC is equal to zero (otherwise, the file's catalog entry is listed):

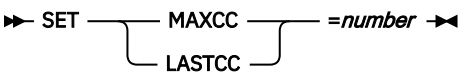
```
IF LASTCC = 0  -
  THEN DO;
    PRINT INFILE (AJK006)
    IF MAXCC=0 -
      THEN REPRO INFILE (AJK006) OUTFILE (AJK007)
    END;
  ELSE LISTCAT ENTRY (FILE006)
```

Notice that a null ELSE clause is not required preceding the END command; it is assumed because of the presence of END.

SET

The SET command is used to change or reset a previously defined condition code. See “IF-THEN-ELSE” on page 201 for the meaning of condition codes.

The format of the SET command is:



where:

SET
specifies that a condition code is to be set. A SET command that follows a THEN or ELSE clause that is not executed does not change or reset a condition code.

MAXCC
specifies that the maximum condition code set by a previous functional command is to be reset. Setting MAXCC does not affect the value of LASTCC even if LASTCC was the maximum condition code set previously.

LASTCC
specifies that the condition code set by the immediately previous functional command is to be reset.

The symbol EQ may be used instead of the = sign.

number - specifies the value to be assigned to MAXCC or LASTCC. The maximum value that can be assigned is 16; a greater value will be reduced to 16. If the value assigned to LASTCC is greater than the value of MAXCC, MAXCC is set equal to LASTCC. All processing terminates when MAXCC or LASTCC is set to 16.

Command Examples:

If you want to set the last condition code established to 12, specify:

```
SET LASTCC=12
```

If you want to replace the highest condition code established so far in processing with 8, specify:

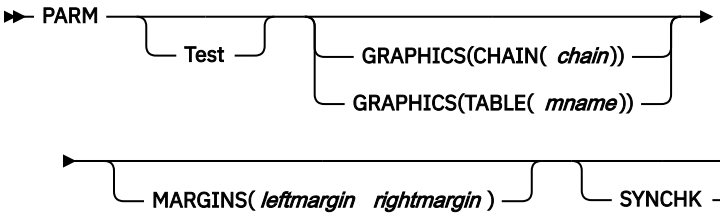
```
SET MAXCC=8
```

PARM

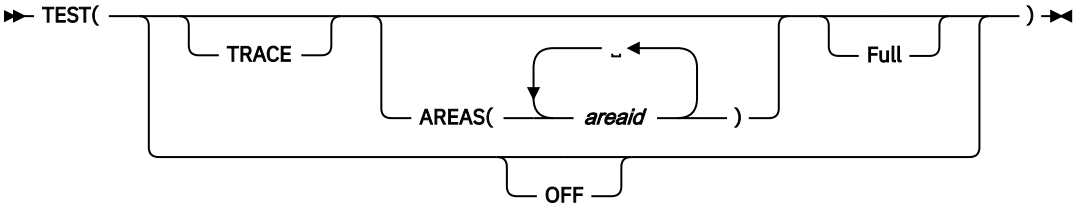
The PARM command specifies processing options for diagnosis tools, syntax checking, and printed output to be used during execution. These options remain in effect until changed by another PARM command.

There are two ways to specify PARM options. One is to run the PARM command, specifying the options. For an example, refer to “Example 25: IDCAMS PRINT That Allows Printing in Upper and Lower Case” on page 239. The other way is to specify the options in the PARM parameter of the EXEC job control statement; for information on how to code the parameter, refer to the "// EXEC Statement" in the VSE/VSAM User's Guide and Application Programming.

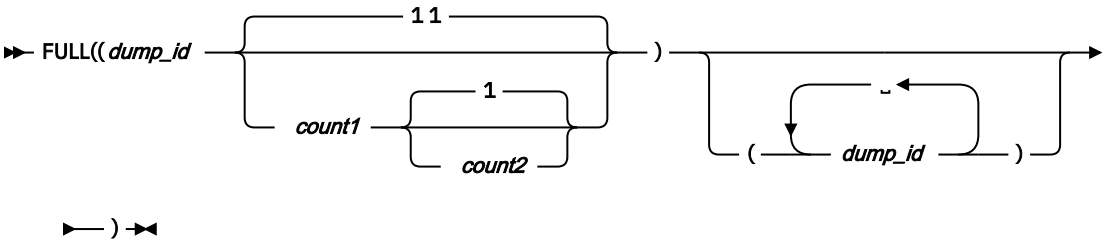
The format of the PARM command is:



Test



Full



where:

AREAS(areaid)
identifies modules that are to have selected variables dumped at their dump points. Every area ID is a two character identifier defined within the implementation.

FULL(dump-id)
specifies that a partition dump, as well as the trace tables and selected variables, are to be provided at the specified points.

dump-id is the four-character ID of the dump point.

count1 is a decimal integer that specifies the number of times the program is to encounter the dump point before beginning the dump listing.

count2 is a decimal integer that specifies the number of encounters at a dump point for which dumps are to be listed.

Default: The default for count1 and count2 is 1.

Restrictions:
count1 and count2 must range between a minimum of 1 and a maximum of 32,767.

GRAPHICS(CHAIN(chain)) | **(TABLE(mname))**
specifies the print chain/train graphic character set or a special graphics table to be used in producing the output.

Abbreviation: GRPH

CHAIN(AN|HN|PN|QN|RN|SN|TN)
specifies the graphic character set of the print chain or train.

Default: PN

TABLE(mname)
specifies the name of a module (accessible through VSE/VSAM CDLOAD) in the sublibrary that contains a 256-byte user-provided translate table. This table defines the graphic characters for each of the possible 256 bit patterns. Any character to be printed is translated to the bit pattern found in such a table at the position corresponding to its numeric value (0-255).

Restriction:
If you specify the TABLE parameter, issue the PARM command early in the IDCAMS command stream to avoid having CDLOAD fail.

MARGINS(leftmargin rightmargin)
specifies that the margins of input records on which command statements and comments are written are to be changed. The standard left and right margins are 2 and 72, respectively. If MARGINS is coded, all subsequent input records are scanned in accordance with the new margins. MARGINS may be used to cause commands coded like comments (between /* and */) to be active or inactive: respecification of margins could be used to cause the /* and */ characters to be omitted from the scan or included.

leftmargin - specifies the left margin.

rightmargin - specifies the right margin.

Abbreviation: MAR

Restriction: The right margin value must be greater than the left margin value.

OFF
specifies that use of diagnosis tools is to stop.

TEST(options)
specifies the diagnosis tools to be used. Once the TEST option has been established, it remains in effect until end of job step or until it is reset by another PARM command. The TRACE, AREAS, and FULL parameters may be used concurrently.

TRACE
specifies that trace tables are to be listed whenever the built-in dump points of the processor are encountered.

SYNCHK
specifies that syntax checking is to be performed on the IDCAMS commands in this job stream. No commands (except PARM) will be executed, and no data will be modified. You may have to run this job several times because VSE/VSAM only detects one error per command in one run.

You should syntax check new command streams before running them for the following reasons:

- Your data is protected because no data is modified if the command does not run correctly.
- You do not have to have a catalog or VSE/VSAM files mounted to run the syntax check.

Messages will indicate the progress of the syntax check. Once messages indicate that no errors are present, run the job without specifying SYNCHK.

Abbreviation: None

Modal: PARM

Chapter 3. Sample IDCAMS Command Job Streams

The following sample job streams are intended to be used only as a guide for the coding of your own job streams. They are not intended to be coded exactly as shown and used by an installation, because the user data that is necessary to run the jobs is not provided.

Note: No DLBL statement for master catalog AMASTCAT appears in the following examples because it is assumed that the following required DLBL statement has been placed in the permanent label area (assume 20 tracks per cylinder):

```
// DLBL IJSYSCT,'AMASTCAT',,VSAM
```

General Examples

The examples in this section illustrate options other than BACKUP and RESTORE, which are treated individually in later sections.

Example 1: Define a System's Catalogs

Before any jobs which use VSE/VSAM can be run, the master catalog must be defined. In this example, the master catalog and one user catalog are defined. Both catalogs are password-protected. The data space for every catalog is defined to be 15 cylinders. The amount of space explicitly specified for the data and index components of every catalog is taken from its respective catalog data space. Because the data space for both the data and index components is less than the total catalog data space, the remaining catalog data space is available for suballocation. The master catalog resides on volume VSER01 and the user catalog on volume VSER02. These volumes may be referenced by other VSE/VSAM catalogs, though they are connected to the catalog which they contain.

In the DEFINE command for D27UCAT1, a data space of 15 cylinders is defined and assigned to class 7 data space. The user catalog is immediately suballocated 4 of the cylinders in this class 7 space; the remaining 11 cylinders of class 7 space is available for later suballocation.

```
// JOB      EXAMPL01
// EXEC     IDCAMS,SIZE=AUTO
   DEFINE   MCAT                        -
              ( NAME(AMASTCAT)          -
                MRPW(MCATMRPW)          -
                UPDPW(MCATUPPW)         -
                RDPW(MCATRDPW)          -
                TO(99366)               -
                ATT(3)                  -
                CODE(BEQUIET)           -
                CYL(15)                 -
                ORIGIN(30)              -
                VOL(VSER01)             -
              )                         -
               DATA                     -
               ( CYL(5 1) )             -
               INDEX                    -
               ( CYL(2) )
```

```
IF      LASTCC = 0                    -
          THEN                        -
            DEFINE  UCAT              -
              ( NAME(D27UCAT1)        -
                MRPW(UCATMRPW)        -
                UPDPW(UCATUPPW)       -
                RDPW(UCATRDPW)        -
                FOR(365)              -
                DEDICATE              -
                VOL(VSER02)           -
                CLASS(7)              -
             )                        -
              DATA                    -
```

```
                  ( CYL(3,1) )          -
                INDEX                   -
                  ( CYL(1) )            -
                CATALOG(AMASTCAT/MCATUPPW)
/*
/&
```

The first DEFINE command defines the master catalog for the system.

1. The MCAT parameter is required and NAME specifies the master catalog being defined.
2. The MRPW parameter specifies the master password of this catalog.
3. The UPDPW parameter specifies the update password of this catalog.
4. The RDPW parameter specifies the read password of this catalog.
5. The TO parameter specifies the maximum retention time.
6. The ATT parameter specifies the number of allowable attempts for catalog password prompting.
7. The CODE parameter specifies the code for password prompting.
8. The CYL parameter specifies the amount of space to be allocated to the catalog's data space. A space parameter is required.
9. The ORIGIN parameter specifies the beginning track number.

   Note: TRK(225) is equivalent to CYL(15).
10. The VOL parameter is required and specifies the volume containing this catalog.
11. The DATA parameter specifies the amount of space to be allocated to the catalog's data component.
12. The INDEX parameter specifies the amount of space to be allocated to the catalog's index component. VSE/VSAM adds together the amount of space specified via the DATA and INDEX parameters and determines the appropriate proportions for every component.

If the definition of the master catalog was successful, a second DEFINE command defines a user catalog that resides on volume VSER02.

1. The UCAT parameter is required and NAME specifies the user catalog being defined.
2. The MRPW parameter specifies the master password for this catalog.
3. The UPDPW parameter specifies the update password of this catalog.
4. The RDPW parameter specifies the read password of this catalog.
5. The FOR parameter specifies the retention period for this file—in this case, one year.
6. The DEDICATE parameter specifies that all unused space in the VTOC is to be allocated to VSE/VSAM.
7. The VOL parameter is required and specifies the volume containing this catalog.
8. The CLASS parameter assigns the data space (specified in the CYL parameter) to class 7. The user catalog is suballocated from this space.
9. The DATA parameter specifies the amount of space to be allocated to the catalog's data component.
10. The INDEX parameter specifies the amount of space to be allocated to the catalog's index component. VSE/VSAM adds together the amount of space specified via the DATA and INDEX parameters and determines the appropriate proportions for every component.
11. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog.

Example 2: Define a VSE/VSAM User Catalog and a VSE/VSAM Data Space

This example defines a user catalog by using the previously-defined user catalog, D27UCAT1, as a model. The second DEFINE command defines VSE/VSAM data space on a volume owned by the master catalog. This allows files to be suballocated onto this volume.

This example can be viewed as a logical follow-on to the previous example.

```
// JOB     EXAMPL02
// EXEC    IDCAMS,SIZE=AUTO
   DEFINE  UCAT                                -
           ( NAME(D27UCAT2)                    -
             MODEL(D27UCAT1/UCATMRPW D27UCAT1) -
             ORIGIN(15)                        -
             VOL(VSER03)                       -
             CYL(5,5)                          -
             CLASS(0)                          -
           )                                   -
           CATALOG(AMASTCAT/MCATUPPW)
    DEFINE  SPACE                              -
           ( ORIGIN(75)                        -
             VOL(VSER03)                       -
             TRK(57)                           -
             CLASS(1)                          -
           )                                   -
           CATALOG(AMASTCAT/MCATUPPW)
```

```
    DEFINE  SPACE                              -
            ( ORIGIN(135)                      -
              VOL(VSER03)                      -
              CYL(9,1)                         -
            )                                  -
           CATALOG(AMASTCAT/MCATUPPW)

/*
/&
```

The first DEFINE command defines a new user catalog modeled with the same self-describing attributes of the user catalog defined in Example 1. The VOL and space (CYL) parameters are required even though modeling is performed.

1. The UCAT parameter is required and NAME specifies the user catalog being defined.
2. The MODEL parameter specifies the name of the catalog to be modeled and the master password of that catalog so as to enable the catalog's security attributes to be modeled. Any parameter that is specified in the modeled catalog and is not overridden here will be used (for example, the retention period will be 365 days).
3. The ORIGIN parameter indicates that 15 is the beginning track number.
4. The VOL parameter is required and specifies the volume containing this catalog.
5. The CYL parameter specifies the amount of space to be allocated to the catalog's data space beginning at the location specified by ORIGIN. A space parameter is required.
6. The CLASS parameter specifies the space class for the new user catalog. The useclass for the catalog is always the same as the class specified (in this case, 0), overriding the useclass 7 of the model. If the class were not explicitly specified, the model useclass would be used for both the class of the space defined and the useclass of the catalog.
7. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog.

The second DEFINE command defines a VSE/VSAM data space on volume VSER03. The space is to be assigned to class 1. Class 1 space is normally assigned to the fixed head area of a direct-access storage device, although this is not a requirement. In this example, the device type is not indicated. Therefore, no attempt has been made to align the class 1 with the fixed head area.

Note: This space belongs to the master catalog and user catalog D27UCAT1 already resides on VSER03. This is supported by VSE/VSAM Version 1 Release 3 and above.

1. The SPACE parameter is required.
2. The ORIGIN parameter indicates that 75 is the beginning track number.
3. The VOL parameter is required and specifies the volume containing this data space.
4. The TRK parameter specifies the amount of space to be allocated on this volume. A space parameter is required.

5. The CLASS parameter specifies that the space is to be assigned to class 1.
6. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog.

The third DEFINE command defines a second VSE/VSAM space on VSER03, making possible the suballocation of files onto this volume. The space is assigned to class 0 by default.

1. The SPACE parameter is required.
2. The ORIGIN parameter indicates that 135 is the beginning track number.
3. The VOL parameter is required and specifies the volume containing this data space.
4. The CYL parameter specifies the amount of space to be allocated on the volume. The space is assigned to class 0 by default. A space parameter is required.
5. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog.

Example 3: Define VSE/VSAM Files

This example defines a unique key-sequenced file into the master catalog, a suballocated relative-record file into the master catalog, and a suballocated entry-sequenced file into the user catalog D27UCAT2. Execution of the LISTCAT commands is dependent upon successful definition of the specified file in the preceding DEFINE command.

Example 3 can be viewed as a logical follow-on to the previous examples.

```
// JOB      EXAMPL03
// DLBL     IJSYSUC,'D27UCAT2',,VSAM
// DLBL     DFILE,,,VSAM
// EXTENT   ,VSER03,1,0,270,30
// DLBL     XFILE,,,VSAM
// EXTENT   ,VSER03,1,0,300,15
// EXEC     IDCAMS,SIZE=AUTO
   DEFINE   CLUSTER                              -
            ( NAME(EXAMPLE.KSDS1)                -
              RDPW(KSD1PSWD)                     -
              FOR(365)                           -
              VOL(VSER03)                        -
              BUFSP(12288)                       -
              RECORDSIZE(400 475)                -
              KEYS(12 4)                         -
              FREESPACE(40 40)                   -
            )                                    -
            DATA                                 -
            ( NAME(EXAMPLE.KSDS1.DATA)           -
              UNIQUE                             -
              CYLINDERS(2 1)                     -
              FILE(DFILE)                        -
            )                                    -
            INDEX                                -
            ( NAME(EXAMPLE.KSDS1.INDEX)          -
              UNIQUE                             -
              FILE(XFILE)                        -
              CYLINDERS(1 1)                     -
            )                                    -
            CATALOG(AMASTCAT/MCATUPPW)
   IF       LASTCC = 0                           -
              THEN                               -
   LISTCAT  ENTRIES(EXAMPLE.KSDS1/KSD1PSWD) ALL -
                       CATALOG(AMASTCAT)
```

```
   DEFINE   CLUSTER                              -
            ( NAME(EXAMPLE.RRDS1)                -
              VOL(VSER03)                        -
              TRK(10 5)                          -
              RECORDSIZE(100 100)                -
              NUMBERED                           -
            )                                    -
            CATALOG(AMASTCAT/MCATUPPW)
   IF       LASTCC=0                             -
```

```
              THEN                                -
    LISTCAT   ENTRIES(EXAMPLE.RRDS1) CLUSTER ALL  -
                        CATALOG(AMASTCAT)
    DEFINE    CLUSTER                             -
              ( NAME(EXAMPLE.ESDS1)               -
                MRPW(ESD1MRPW)                    -
                UPDPW(ESD1UPPW)                   -
                VOL(VSER03)                       -
                SPANNED                           -
                REUSE                             -
                NONINDEXED                        -
                CYL(1 1)                          -
                RECORDSIZE(2500 3000)             -
              )                                   -
              CATALOG(D27UCAT2/UCATMRPW)
    IF        LASTCC = 0                          -
              THEN                                -
    LISTC     ENTRIES(EXAMPLE.ESDS1) ALLOC
  /*
  /&
```

Explanation of JCL

1. The IJSYSUC DLBL statement describes the user catalog into which the entry-sequenced file is to be defined. IJSYSUC is used to designate a user catalog as a job catalog. All references will be to the job catalog unless otherwise directed.
2. The DFILE and XFILE DLBL statements describe the data components of the data and index components, respectively, of the unique key-sequenced file being defined and their corresponding EXTENT statements specify the area of disk allocation to be assigned to every component. Note that the EXTENT statements do not require a symbolic unit specification.

Explanation of the IDCAMS Commands

The first DEFINE command defines a unique key-sequenced file on volume VSER03.

1. The CLUSTER parameter is required and NAME specifies the cluster being defined.
2. The RDPW parameter specifies the read password of this cluster. Because no master password is defined, this password will be propagated up to the master level.
3. The FOR parameter specifies the retention period for this file—in this case, one year.
4. The VOL parameter is required and specifies the volume containing this file.
5. The BUFSP parameter is specified for increased performance.
6. The RECORDSIZE parameter specifies the average and maximum record sizes.
7. The KEYS parameter specifies the key length and offset from the beginning of the record.
8. The FREESPACE parameter specifies the percentage of space within CIs and CAs, respectively, that is originally free.
9. The DATA parameter is required if attributes are to be specified for the data component. The NAME parameter specifies the name of the data component. If a name is not specified, a name is generated.
10. The UNIQUE parameter specifies that this portion of the file is the only one that occupies the data space allocated to it.
11. A space parameter (in this example CYLINDERS) is required for a unique file. The amount of space specified to be allocated to the data component must match the space parameters in the corresponding EXTENT statement. The secondary allocation quantity is ignored.
12. The FILE parameter is required for a unique file and names the filename of the DLBL statement for the data component.
13. The INDEX parameter is required if attributes are to be specified for the index component. The NAME parameter specifies the name of the index component. If a name is not specified, a name is generated.
14. The UNIQUE parameter specifies that this portion of the file is the only one that occupies the data space allocated to it.

15. The FILE parameter is required for a unique file, and gives the filename of the DLBL statement for the index component.
16. A space parameter, in this example CYLINDERS, is required for a unique file and the amount of space specified to be allocated to the index component must match the space parameters in the corresponding EXTENT statement. The secondary allocation quantity is ignored.
17. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog. This parameter is also required because a DLBL statement for the job catalog (IJSYSUC) appears in the JCL and this DEFINE must therefore be explicitly directed to the master catalog.

If the DEFINE of the unique key-sequenced file was successful, then the following LISTCAT command is executed.

1. The ENTRIES and ALL parameters cause the entire catalog description of the file just defined to be listed.
2. The CATALOG parameter directs the LISTCAT to the master catalog.

The second DEFINE command defines a suballocated relative-record file into the VSE/VSAM class 0 data space on volume VSER03 which is owned by the master catalog. Class 0 is selected by default because of the absence of the USECLASS parameter.

1. The CLUSTER parameter is required and NAME specifies the cluster being defined.
2. The VOL parameter is required and specifies the volume containing this file.
3. The TRK parameter specifies the amount of space allocated to this file. A space parameter is required.
4. The RECORDSIZE parameter specifies the average and maximum record sizes which, in the case of a relative-record file, must be equal.
5. The NUMBERED parameter which specifies a relative record file is required to override the default (INDEXED).
6. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog. This parameter is also required because a DLBL statement for the job catalog (IJSYSUC) appears in the job control and this define is directed to the master catalog.

If the DEFINE of the suballocated relative-record file was successful, then the following LISTCAT command is executed.

1. The ENTRIES, CLUSTER and ALL parameters cause the entry just defined to be listed, limited, however, to the cluster object.
2. The CATALOG parameter directs the LISTCAT to the master catalog.

The third DEFINE command defines a suballocated entry-sequenced file into the job catalog (IJSYSUC). Note that no read password was specified so that the cluster can be read without a password.

1. The CLUSTER parameter is required and NAME specifies the cluster being defined.
2. The MRPW parameter specifies the master password of this cluster.
3. The UPDPW parameter specifies the update password of this cluster.
4. The VOL parameter is required and specifies the volume containing this file.
5. The SPANNED parameter specifies that records may span control interval boundaries.
6. The REUSE parameter specifies that the file can be reused, that is, reloaded without being deleted and redefined.
7. The NONINDEXED parameter which specifies entry sequence is required to override the default (INDEXED).
8. The CYL parameter specifies the amount of space to be allocated to this file. Space is allocated from class 0 space because of the absence of the USECLASS parameter. A space parameter is required.
9. The RECORDSIZE parameter specifies the average and maximum record sizes.

10. The CATALOG parameter specifies the *file ID* of the job catalog. It is required only because the catalog is password-protected. Either the master or update password is required. The master password is shown here.

If the DEFINE of the suballocated entry-sequenced file was successful, then the following LISTCAT command is executed.

1. The ENTRIES and ALLOC parameters cause the file entry just defined to be listed, limited, however, to only volume and allocation information.
2. The absence of a CATALOG parameter implicitly directs the LISTCAT to the job catalog (IJSYSUC).

Example 4: Define NonVSAM and VSE/VSAM Files

This example defines a nonVSAM file into a user catalog, and VSE/VSAM key-sequenced and entry-sequenced files into the master catalog. Attributes for the entry-sequenced file are modeled from those of the entry-sequenced file defined in Example 3.

Example 4 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL04
// DLBL    IJSYSUC,'D27UCAT2',,VSAM
// EXEC    IDCAMS,SIZE=AUTO
   DEFINE  NONVSAM                                -
           ( NAME(EXAMPLE.NONVSAM1)               -
             VOL(VSER02)                          -
             DEVT(3380)                           -
           )                                      -
           CATALOG (D27UCAT1/UCATMRPW)
   IF      LASTCC = 0                             -
             THEN                                 -
   LISTCAT  NONVSAM ALL                           -
            CATALOG(D27UCAT1)
```

```
      DEFINE   CLUSTER                            -
               ( NAME(EXAMPLE.KSDS2)              -
                 MRPW(KSD2MRPW)                   -
                 UPDPW(KSD2UPPW)                  -
                 RDPW(KSD2RDPW)                   -
               )                                  -
               DATA                               -
               ( NAME(EXAMPLE.KSDS2.DATA)         -
                 RECORDS(500 100)                 -
                 USECLASS(0 0)                    -
                 EXCEPTIONEXIT(DATEXIT)           -
                 ERASE                            -
                 FREESPACE(20 10)                 -
                 KEYS(6 4)                        -
                 RECORDSIZE(80 100)               -
                 ORDERED                          -
                 VOL(VSER03 VSER01)               -
               )                                  -
               INDEX                              -
               ( NAME(EXAMPLE.KSDS2.INDEX)        -
                 RECORDS(100 100)                 -
                 USECLASS(1 P)                    -
                 VOL(VSER03)                      -
               )                                  -
               CATALOG(AMASTCAT/MCATUPPW)
      IF       LASTCC = 0                         -
                 THEN                             -
      LISTCAT  DATA ALL ENT(EXAMPLE.KSDS2/KSD2MRPW)  -
               CATALOG(AMASTCAT)
```

```
      DEFINE   CLUSTER                            -
               ( NAME(EXAMPLE.ESDS2)              -
                 MODEL(EXAMPLE.ESDS1/ESD1MRPW)    -
                 VOLUMES(VSER03)                  -
                 USECLASS(1 0)                    -
                 MRPW(ESD2MRPW)                   -
                 CTLPW(ESD2CTPW)                  -
                 UPDPW(ESD2UPPW)                  -
                 RDPW(ESD2RDPW)                   -
               )                                  -
               CATALOG(AMASTCAT/MCATUPPW)
```

```
      IF       LASTCC = 0                              -
                THEN                                   -
                 DO
                   LISTC ENT(EXAMPLE.ESDS2/ESD2MRPW)   -
                         ALL                           -
                         CATALOG(AMASTCAT)
                   LISTC NAME CATALOG(AMASTCAT/MCATMRPW)
                 END
   /*
   /&
```

Explanation of JCL

1. The IJSYSUC DLBL statement describes the user catalog D27UCAT2 as a job catalog. It contains the model to be used when defining the entry-sequenced file in this example. All references will be to this job catalog unless otherwise directed.

Explanation of the IDCAMS Commands

The first DEFINE command defines an existing nonVSAM file into user catalog D27UCAT1.

1. The NONVSAM parameter is required and NAME specifies the nonVSAM object being defined.
2. The VOL parameter is required and specifies the volume containing the file.
3. The DEVT parameter is required and specifies the device type of the volume.
4. The CATALOG parameter specifying the name of the user catalog is required because:
   - A job catalog also appears in the job control so the parameter must explicitly direct the DEFINE to the user catalog.
   - The catalog is protected and its master password is needed for the DEFINE.

If the DEFINE of the nonVSAM entry was successful, then the following LISTCAT command is executed.

1. The NONVSAM and ALL parameters cause all the nonVSAM entries cataloged in D27UCAT1 to be listed.
2. The CATALOG parameter directs the LISTCAT to a specific user catalog and opens that catalog.

The second DEFINE command defines a key-sequenced file into VSE/VSAM data space owned by the master catalog. Note that attributes are specified at the data and index level rather than the cluster level.

1. The CLUSTER parameter is required and NAME specifies the name of the cluster being defined.
2. The MRPW, UPDW, and RDPW parameters specify the master, update, and read passwords, respectively, of the cluster.
3. The DATA component is explicitly named via the NAME parameter; it will not have a name generated for it by VSE/VSAM.
4. The RECORDS parameter specifies the amount of space to be allocated to the data component.
5. The USECLASS parameter specifies that the data component is to be allocated from class 0 data space.
6. The EXCEPTIONEXIT parameter specifies the name of the routine to be given control if an exception occurs while processing the data component.
7. The ERASE parameter specifies that the data component is to be overwritten with binary zeros when it is deleted.
8. The FREESPACE parameter specifies the percentage of space within CIs and CAs, respectively, that is originally free.
9. The KEYS parameter specifies the key length and offset.
10. The RECORDSIZE parameter specifies the average and maximum record sizes.
11. The ORDERED parameter specifies that VSER03 is to be used first (primary allocation). If VSER03 does not have sufficient space for the primary allocation, the DEFINE fails. Volume VSER01 is to

be used only for secondary extensions when class 0 space is not available on VSER03; the first allocation on VSER01 is a primary allocation.

12. The VOL parameter specifies the volume (VSER03) containing this data component and the volume used for extending the file (VSER01).
13. The INDEX component is explicitly named via the NAME parameter; it will not have a name generated for it by VSE/VSAM.
14. The RECORDS parameter specifies the amount of space to be allocated to the index component.
15. The USECLASS parameter specifies that the index component is to be allocated from class 1 data space and secondary extensions are also to be allocated from class 1. (Class 1 space was established on volume VSER03 in Example 2.)
16. The VOL parameter specifies the volume containing this index component.
17. The CATALOG parameter is required because the master catalog is password protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog. This parameter is also required because a DLBL statement for the job catalog (IJSYSUC) appears in the job control and this define must, therefore, be explicitly directed to the master catalog.

If the DEFINE of the key-sequenced file was successful, then the following LISTCAT command is executed.

1. The DATA, ALL, and ENT parameters cause all of the information contained in the data component entry to be listed.
2. The CATALOG parameter directs the LISTCAT to the master catalog.

The third DEFINE command defines an entry-sequenced file that uses the entry-sequenced file (EXAMPLE.ESDS1) defined in Example 3 as a model. The protection attributes of the model, the containing volume, and the class of space to be used are, however, overridden by explicit specification. The define is directed to the master catalog.

1. The CLUSTER parameter is required and NAME specifies the cluster being defined.
2. The MODEL parameter specifies the name of the file to be used as a model. Note that the new entry-sequenced file is being defined into the master catalog, but the entry-sequenced file to be used as a model is defined in job catalog D27UCAT2.
3. The VOLUMES parameter specifies a volume (VSER03). It is the same volume on which the user catalog is defined. This define suballocates into the space that belongs to the master catalog.
4. The USECLASS parameter overrides the model useclass specification of 0 and specifies that the primary allocation for the file is to be taken from class 1 space on volume VSER03. However, secondary extensions are to be allocated from class 0 space.
5. The MRPW, CTLPW, UPDPW, and RDPW parameters specify passwords different from the passwords specified for the file being modeled.
6. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog. This parameter is also required because a DLBL statement for the job catalog (IJSYSUC) appears in the job control and this define must, therefore, be explicitly directed to the master catalog.

If the DEFINE of the entry-sequenced file was successful, then the following LISTCAT commands are executed.

1. The ENT and ALL parameters of the first LISTC command cause all the cataloged information in the entry just defined to be listed.
2. The CATALOG parameter directs the LISTCAT to the master catalog.
3. The NAME parameter of the second LISTC command causes only the names of the objects cataloged in the master catalog to be listed.
4. The CATALOG parameter directs the LISTCAT to the master catalog.

Example 5: Loading and Printing of Files

This example shows various techniques which can be used to load and print files using the REPRO and PRINT commands.

Example 5 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL05
// ASSGN   SYS013,3380,VOL=VSER03
*          NONVSAM INDEXED SEQUENTIAL FILE
// DLBL    INDSET1,'BADVA101.TEST.ISAM1',,ISE
// EXTENT  SYS013,VSER03,4,0,375,15
// EXTENT  SYS013,VSER03,4,1,390,15
// EXTENT  SYS013,VSER03,1,2,405,15
*          NONVSAM VARIABLE SAM FILE
// DLBL    INDSET2,'BADVA101.TEST.SAM2'
// EXTENT  SYS013,VSER03,1,0,420,5
*          NONVSAM FIXED SAM FILE
// DLBL    INDSET3,'BADVA101.TEST.SAM1'
// EXTENT  SYS013,VSER03,1,0,425,5
*          NONVSAM FIXED SAM FILE
// DLBL    INDSET4,'EXAMPLE.NONVSAM1'
// EXTENT  SYS013,VSER03,1,0,430,15
*          VSE/VSAM FILES
// DLBL    VSDSET1,'EXAMPLE.KSDS1',,VSAM
// DLBL    VSDSET2,'EXAMPLE.ESDS1',,VSAM,CAT=UCAT2
// DLBL    VSDSET3,'EXAMPLE.RRDS1',,VSAM
// DLBL    VSDSET4,'EXAMPLE.KSDS2',,VSAM
// DLBL    UCAT2,'D27UCAT2',,VSAM
// EXEC    IDCAMS,SIZE=AUTO
          /* LOAD A VSE/VSAM KEY-SEQUENCED FILE   */
          /* FROM AN ISAM FILE                 */
   REPRO   INFILE (INDSET1 ENV(RECFM(F)         -
                  BLKSZ(100) RECSZ(100)))       -
            OUTFILE (VSDSET1/KSD1PSWD)
   IF       MAXCC = 0                           -
              THEN -
              PRINT  INFILE(VSDSET1/KSD1PSWD)   -
                            FROMKEY(ABC)
   /* LOAD A VSE/VSAM ENTRY-SEQUENCED FILE FROM AN */
   /* EXISTING VARIABLE UNBLOCKED SAM FILE     */
   REPRO   INFILE (INDSET2 ENV(RECFM(V)         -
                  RECSZ(132)                    -
                  BLKSZ(136)))                  -
           OUTFILE (VSDSET2/ESD1UPPW)           -
                    COUNT(30)
   IF      LASTCC = 0                           -
             THEN                               -
             PRINT  INFILE (VSDSET2/ESD1PSWD)   -
                           FADDR(65)            -
                           HEX
   /* LOAD A VSE/VSAM RELATIVE RECORD FILE         */
   /* FROM AN EXISTING SAM FILE                */
   REPRO     INFILE (INDSET3  ENV(RECFM(F)      -
                     BLKSZ(100)))               -
             OUTFILE (VSDSET3)                  -
                      SKIP(10)
```

```
   IF      LASTCC = 0                           -
           THEN                                 -
           PRINT  INFILE (VSDSET3)              -
                          TNUM(25)
   /* PRINT THE CONTENTS OF THE SAM FILE       */
   PRINT INFILE (INDSET3 ENV(RECFM(F)           -
                 BLKSZ(100)))                   -
         COUNT(20)                              -
         CHAR
   /* LOAD A VSE/VSAM SEQUENCED FILE               */
   /* FROM A NONVSAM FILE                      */
   REPRO   INFILE (INDSET4 ENV(RECFM(F)         -
                   BLKSZ(80)))                  -
           OUTFILE (VSDSET4/KSD2UPPW)
   IF      LASTCC = 0                           -
            THEN                                -
              PRINT INFILE (VSDSET4/KSD2RDPW)   -
                            FROMKEY(AAAAJA)     -
                            TOKEY(AAAAJ9)
```

```
/*
/&
```

Explanation of JCL

1. The INDSET1 DLBL statement describes the ISAM file which is to be copied into the VSE/VSAM file ‘EXAMPLE.KSDS1’.
2. The INDSET2 DLBL statement describes the variable-length SAM file which is to be copied into the VSE/VSAM file ‘EXAMPLE.ESDS1’.
3. The INDSET3 DLBL statement describes the fixed SAM file which is to be copied into the VSE/VSAM file ‘EXAMPLE.RRDS1'.
4. The INDSET4 DLBL statement describes the fixed SAM file which is to be copied into the VSE/VSAM file ‘EXAMPLE.KSDS2'.
5. The VSDSET1 DLBL statement describes a VSE/VSAM key-sequenced file which is to be loaded and printed.
6. The VSDSET2 DLBL statement describes a VSE/VSAM entry-sequenced file which is to be loaded and printed and indicates that it is in the catalog identified by the UCAT2 DLBL statement.
7. The VSDSET3 DLBL statement describes a VSE/VSAM relative-record file which is to be loaded and printed.
8. The VSDSET4 DLBL statement describes a VSE/VSAM key-sequenced file which is to be loaded and printed.
9. The UCAT2 DLBL statement describes the user catalog in which the file ‘EXAMPLE.ESDS1' is cataloged. Specification of the user catalog by the CAT parameter on the DLBL statement with filename VSDSET2 enables reference to cataloged information describing the file which is required to open it. The VSE/VSAM master catalog need *not* be specified because it can always be referenced in the absence of a DLBL statement for the job catalog (IJSYSUC).

Explanation of the IDCAMS Commands

The first REPRO command causes a VSE/VSAM key-sequenced file to be loaded from an ISAM file.

1. The INFILE parameter is required and names the file containing the source data. The filename of the DLBL statement for this file must be identical to this name. The ENV subparameter is required for all nonVSAM files.
2. The OUTFILE parameter is required and names the target file. The filename of the DLBL statement for this file must be identical to this name. The update or higher password of the VSE/VSAM file is required.

If the REPRO operation was successfully executed, then the contents of the VSE/VSAM key-sequenced file just loaded is printed. The format of the listing is DUMP, because the default is taken.

1. The INFILE parameter is required and names the file to be printed. The filename of the DLBL statement for this file must be identical to this name. The read or higher password is required to print the file.
2. The FROMKEY parameter specifies that printing is to begin with the record whose key (high order three bytes) is greater than or equal to ‘ABC'.

The second REPRO command causes a VSE/VSAM entry-sequenced file to be loaded from an existing variable unblocked SAM file.

1. The INFILE parameter is required and names the file containing the source data. The filename of the DLBL statement for this file must be identical to this name. The ENV subparameter is required for all nonVSAM files. For a variable SAM file, BLKSZ must equal the largest block, and RECSZ must be 4 less than BLKSZ.
2. The OUTFILE parameter is required and names the target file. The filename of the DLBL statement for this file must be identical to this name. The update or higher password of the VSE/VSAM file is required.

3. The COUNT parameter specifies that the first 30 records of the SAM file are to be loaded.

If the REPRO operation was successfully executed, then the contents of the VSE/VSAM entry-sequenced file just loaded is printed in hexadecimal format.

1. The INFILE parameter is required and names the file to be printed. The filename of the DLBL statement for this file must be identical to this name. Note that a password is *not* required, because the file does not have a read password.
2. The FADDR parameter specifies that the first record printed is that record whose relative byte address is exactly equal to 65.
3. The HEX parameter specifies that the listing is to be hexadecimal format.

The third REPRO command causes a VSE/VSAM relative-record file to be loaded from an existing fixed SAM file. The relative-record file can receive only fixed length records that equal its defined record length.

1. The INFILE parameter is required and names the file containing the source data. The filename of the DLBL statement for this file must be identical to this name. The ENV subparameter is required for all nonVSAM files.
2. The OUTFILE parameter is required and names the target file which is not password-protected. The filename of the DLBL statement for this file must be identical to this name.
3. The SKIP parameter specifies that the first 10 records of the SAM file are to be bypassed.

If the REPRO operation was successfully executed, then the contents of the VSE/VSAM relative-record file just loaded are printed.

1. The INFILE parameter is required and names the file to be printed. The filename of the DLBL statement for this file must be identical to this name. A password is not necessary when a file is not password-protected.
2. The TNUM parameter limits the output to the first 25 relative records.

The PRINT command causes the contents of the SAM file to be printed.

1. The INFILE parameter is required and names the file to be printed. The filename of the DLBL statement for this file must be identical to this name. The ENV subparameter is required for all nonVSAM files.
2. The COUNT parameter specifies that only 20 records are to be printed.
3. The CHAR parameter specifies that the listing is to be character format.

The last REPRO command causes a VSE/VSAM key-sequenced file to be loaded from a nonVSAM file.

1. The INFILE parameter is required and names the file containing the source data. The filename of the DLBL statement for this file must be identical to this name. The ENV subparameter is required for all nonVSAM files.
2. The OUTFILE parameter is required and names the target file. The filename of the DLBL statement for this file must be identical to this name. Because the VSE/VSAM file is password protected, its update or higher-level password is required.

If the REPRO operation was successfully executed, then the contents of the VSE/VSAM key-sequenced file just loaded are printed. The FROMKEY and TOKEY parameters are used to limit the output to a specific range of keys.

1. The INFILE parameter is required and names the file containing the source data. The filename of the DLBL statement for this file must be identical to this name. Because the VSE/VSAM file is password-protected, its read or higher-level password is required.
2. The FROMKEY and TOKEY parameters specify the keys at which printing is to begin and end, respectively.

Example 6: Modifying and Printing the Contents of VSE/VSAM Files

This example shows techniques for modifying the contents of VSE/VSAM files using the REPRO command. It can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL06
// ASSGN   SYS013,3380,VOL=VSER03
// DLBL    UCAT2,'D27UCAT2',,VSAM
// DLBL    INDSET4,'EXAMPLE.NONVSAM2'
// EXTENT  SYS013,VSER03,1,0,445,5
// DLBL    VSDSET2,'EXAMPLE.ESDS1',,VSAM,CAT=UCAT2
// DLBL    VSDSET3,'EXAMPLE.RRDS1',,VSAM
// DLBL    VSDSET4,'EXAMPLE.KSDS2',,VSAM
// EXEC    IDCAMS,SIZE=AUTO,PARM='SYNCHK'
   REPRO   INFILE(INDSET4,                    -
                  ENV(RECFM(F)                -
                      BLKSZ(80)))             -
           OUTFILE(VSDSET4/KSD2UPPW)          -
           REPLACE
   IF      LASTCC = 0                         -
             THEN                             -
             PRINT   INFILE(VSDSET4/KSD2RDPW) -
                            FROMKEY(AAAAJA)   -
                            TOKEY(AAAAJ9)
   REPRO   INFILE(VSDSET3)                    -
           OUTFILE(VSDSET2/ESD1UPPW)          -
           REUSE
   IF      LASTCC = 0                         -
            THEN                              -
              PRINT   INFILE(VSDSET2)
/*
/&
```

Explanation of JCL

1. The UCAT2 DLBL statement describes the user catalog D27UCAT2. This DLBL statement will be explicitly referenced by means of the CAT parameter of the DLBL statement for the file cataloged in D27UCAT2. Because this job contains files cataloged in both the master and user catalogs, a DLBL statement with the filename IJSYSUC cannot be used as there is then no way to explicitly reference the master catalog by means of the REPRO command.
2. The INDSET4 DLBL statement describes the nonVSAM file to be copied into the VSE/VSAM file EXAMPLE.KSDS2.
3. The VSDSET2 DLBL statement describes a VSE/VSAM entry-sequenced file which is to be reloaded and indicates that it is in the catalog identified by the UCAT2 DLBL statement.
4. The VSDSET3 DLBL statement describes a VSE/VSAM relative-record file which is to be copied into the VSE/VSAM entry-sequenced file EXAMPLE.ESDS2.
5. The VSDSET4 DLBL statement describes a VSE/VSAM key-sequenced file which is to receive replacement records.

Explanation of the IDCAMS Commands

The first REPRO command causes records in the VSE/VSAM key-sequenced file to be replaced with input from a nonVSAM file.

1. The INFILE parameter is required and names the file containing the source data. The filename of the DLBL statement for this file must be identical to this name. The ENV subparameter is required for all nonVSAM files.
2. The OUTFILE parameter is required and names the target file. The filename of the DLBL statement for this file must be identical to this name. Because the VSE/VSAM file is password protected, its update or higher-level password is required.
3. The REPLACE parameter causes a record in the output file having the same key as a record in the input file to be replaced. Records in the input file whose key is not already contained in the output file will be inserted in the output file.

If the REPRO operation was successfully executed, then the contents of the VSE/VSAM key-sequenced file just changed are printed.

1. The INFILE parameter is required and names the file containing the source data. The filename of the DLBL statement for this file must be identical to this name. Because the VSE/VSAM file is password-protected, its read or higher-level password is required.
2. The FROMKEY and TOKEY parameters specify the keys at which printing is to begin and end, respectively.

The second REPRO command causes the VSE/VSAM entry-sequenced file to be loaded from the VSE/VSAM relative-record file.

1. The INFILE parameter is required and names the file containing the source data. The filename of the DLBL statement for this file must be identical to this name. Because this file is unprotected, no password is required.
2. The OUTFILE parameter is required and names the target file. The filename of the DLBL statement for this file must be identical to this name. The update or higher password of the VSE/VSAM file is required.
3. The REUSE parameter specifies that any records already in the entry-sequenced file output are to be overwritten (because the entry-sequenced file was defined with the REUSE attribute).

If the REPRO operation was successfully executed, then the entire contents of the reloaded VSE/VSAM entry-sequenced file are printed.

1. The INFILE parameter is required and names the file containing the source data. The filename of the DLBL statement for this file must be identical to this name. Because no read password was specified for this file, no password is required.

Example 7: Modifying and Listing the Cataloged Attributes of a File

This example shows how the cataloged attributes of a file may be modified. It can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL07
// DLBL    IJSYSUC,'D27UCAT2',,VSAM
// EXEC    IDCAMS,SIZE=AUTO
   ALTER   EXAMPLE.KSDS1.DATA/KSD1PSWD        -
           FREESPACE(10,10)                   -
           CATALOG(AMASTCAT/MCATMRPW)
   IF      LASTCC = 0                         -
             THEN                             -
   LISTC   ENT(EXAMPLE.KSDS1.DATA/KSD1PSWD)   -
           ALL                                -
           CATALOG(AMASTCAT/MCATMRPW)
   ALTER   EXAMPLE.ESDS1/ESD1MRPW             -
           MRPW(ESD1PWMR)                     -
           CTLPW(ESD1PWCT)                    -
           UPDPW(ESD1PWUP)                    -
           RDPW(ESD1PWRD)                     -
           AUTH(ESD1AUTH)                     -
           CATALOG(D27UCAT2/UCATMRPW)
   IF      LASTCC = 0                         -
             THEN                             -
   LISTC   ENT(EXAMPLE.ESDS1/ESD1PWMR)        -
           CLUSTER                            -
           ALL                                -
           CATALOG(D27UCAT2/UCATMRPW)
/*
/&
```

Explanation of JCL

1. The IJSYSUC DLBL statement describes the user catalog D27UCAT2 as a job catalog.

Explanation of the IDCAMS Commands

The first ALTER command (of EXAMPLE.KSDS1.DATA) shows a way to tune space management for a file. The data object was originally defined (see Example 3) with 40% free space for the data component. After initial loading, this percentage is reduced because further activity against this file will not be of the mass insert type.

1. The name of the data object is required as all alterations must name the object which contains the attributes to be altered. The cluster password is required to alter the data component.
2. The FREESPACE parameter specifies the new percentages of free space to be allowed in CIs and CAs, respectively.
3. The CATALOG parameter specifies the name of the master catalog. Because a DLBL statement with the filename IJSYSUC appears in the job control, the master catalog must be explicitly specified in order to alter an object cataloged in it. The master password is required because the catalog is password-protected.

If the ALTER operation was successfully executed, then the data object altered is listed.

1. The ENT parameter specifies the name of the data object to be listed.
2. The ALL parameter specifies that all data object fields are to be listed.
3. The CATALOG parameter is required to name the master catalog. Because a DLBL statement with the filename IJSYSUC appears in the job control but the entry is in the master catalog, the master catalog must be specified. The master password is required because LISTCAT ALL displays protection attributes and the entry is password-protected.

The second ALTER command (of EXAMPLE.ESDS1) shows a way to change the security scheme of a file. Establishing a security scheme for an existing nonprotected file would be done in the same manner.

1. The name of the cluster object is required as all alterations must name the object which contains the attributes to be altered. The master password of the cluster is required to alter a security-protected object.
2. The MRPW, CTLPW, UPDPW, and RDPW parameters respectively specify the new master, control, update, and read passwords for the cluster object.
3. The AUTH parameter specifies the name of the user authorization verification module, which is invoked for access authorization of this file when the master password is not provided.
4. The CATALOG parameter is required to specify the master password of the user catalog, even that this user catalog is the job catalog.

If the ALTER operation was successfully executed, then the cluster object just altered is listed.

1. The ENT parameter specifies the name of the cluster object to be listed. The master password of the object allows its security information to be listed.
2. The CLUSTER parameter specifies that only the attributes of the cluster object will be listed.
3. The ALL parameter specifies that all cluster object attributes are to be listed.
4. The CATALOG parameter is required to specify the master password of the user catalog, even that this user catalog is the job catalog.

Example 8: Creating an Alternate Index and Its Path

This example defines an alternate index over a previously loaded VSE/VSAM key-sequenced base cluster, defines a path over the alternate index to provide a means for processing the base through the alternate index, and builds the alternate index. The alternate index, path, and base cluster must all be defined in the same catalog, in this case, the master catalog.

Example 8 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL08
// DLBL    PATHDD,'EXAMPLE.PATH',,VSAM
// EXEC    IDCAMS,SIZE=AUTO
   DEFINE  AIX                                -
```

```
            ( NAME(EXAMPLE.AIX)                -
              RELATE(EXAMPLE.KSDS2/KSD2MRPW)   -
              MRPW(AIXMRPW)                    -
              UPDPW(AIXUPPW)                   -
              READPW(AIXRDPW)                  -
```

```
              KEYS(3 0)                        -
              RECORDSIZE(40 50)                -
              VOL(VSER03)                      -
              CYL(3 1)                         -
              NONUNIQUEKEY                     -
              UPGRADE                          -
            )                                  -
            CATALOG(AMASTCAT/MCATUPPW)
    /* DEFINE A PATH OVER THE ALTERNATE INDEX */
    DEFINE PATH                                -
            ( NAME(EXAMPLE.PATH)               -
              PATHENTRY(EXAMPLE.AIX/AIXMRPW)   -
              READPW(PATHRDPW)                 -
            )                                  -
            CAT(AMASTCAT/MCATUPPW)
    /* BUILD THE ALTERNATE INDEX */
    BLDINDEX                                   -
             INDATASET(EXAMPLE.KSDS2/KSD2RDPW) -
             OUTDATASET(EXAMPLE.AIX/AIXUPPW)   -
             CATALOG(AMASTCAT/MCATUPPW)        -
             WORKVOLUMES(VSER01)
    /* PRINT THE BASE CLUSTER VIA THE ALTERNATE KEY */
    /* USING THE PATH DEFINED TO CREATE THIS        */
    /* RELATIONSHIP                                 */
    PRINT INFILE(PATHDD/PATHRDPW)
/*
/&
```

Explanation of JCL

1. The PATHDD DLBL statement describes the path relating the alternate index and base cluster. It is required for the PRINT command.
2. No job catalog job control is required because all functions will use the master catalog.

Explanation of the IDCAMS Commands

The first DEFINE command creates a VSE/VSAM alternate index over the base cluster EXAMPLE.KSDS2.

1. The NAME parameter is required and names the object being defined.
2. The RELATE parameter is required and specifies the name of the base cluster over which the alternate index is defined and provides the base cluster master password.
3. The MRPW, UPDPW, and READPW parameters specify the master, update, and read passwords, respectively, for the alternate index.
4. The KEYS parameter specifies the length of the alternate key and its offset in the base cluster record.
5. The RECORDSIZE parameter specifies the length of the alternate index record. It must be large enough to contain the prime keys for all occurrences of any one alternate key because the alternate index is being defined with the NONUNIQUEKEY attribute.
6. The VOL parameter specifies the volume containing the alternate index.
7. The CYL parameter specifies the amount of space to be allocated to the alternate index.
8. The NONUNIQUEKEY parameter specifies that the base cluster can contain multiple occurrences of any one alternate key.
9. The UPGRADE parameter specifies that the alternate index is to reflect all changes made to the base cluster records, for example, additions or deletions of base cluster records.
10. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog.

The second DEFINE command defines a path over the alternate index. After the alternate index has been built, opening with the path name will cause processing of the base cluster via the alternate index.

1. The NAME parameter is required and names the object being defined.
2. The PATHENTRY parameter is required and specifies the name of the alternate index over which the path is defined and its master password.
3. The READPW parameter specifies a read password for the path; it will be propagated to master password level.
4. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog.

The BLDINDEX command builds an alternate index.

1. The INDATASET parameter is required and names the base cluster, and supplies the base cluster's read password which is required for defining an alternate index over it.
2. The OUTDATASET parameter is required and names the alternate index. The update password of the alternate index is also required.
3. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password. If it is necessary for BLDINDEX to use external sort work files, they will be defined in and deleted from the master catalog. The update password will permit these actions.
4. The WORKVOLUMES parameter is required except if an entry-sequenced default model exists in the pertinent catalog. Space for work files will be allocated on volume VSER01.

The PRINT command causes the base cluster to be printed by means of the alternate key using the path defined to create this relationship.

1. The INFILE parameter is required and names the path object. The filename of the DLBL statement for this object must be identical to this name. The password required is the read or higher-level password of the object being opened, in this case, the path.

Example 9: Defining a VSE/VSAM Data Space and Cluster on an FBA Volume

This example defines a VSE/VSAM data space and cluster on an FBA volume owned by the master catalog.

Example 9 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPLE09
// EXEC    IDCAMS,SIZE=AUTO
   /* DEFINE A SPACE ON A VOLUME ASSIGNED TO AN FBA DEVICE */
   DEFINE  SPACE                      -
           ( ORIGIN(1600)             -
             VOL(VSER04)              -
             BLOCKS(12000)            -
             CLASS(0) )               -
         CATALOG(AMASTCAT/MCATUPPW)
   /* DEFINE A KEY-SEQUENCED FILE IN THE CLASS 0 SPACE */
   /* ON THE FBA VOLUME                                */
   DEFINE CLUSTER                     -
          ( NAME(EXAMPLE.KSDS3)       -
            RDPW(KSD3PSWD)            -
            VOL(VSER04)               -
            RECORDSIZE(250,250)       -
            KEYS(8 2)                 -
            FREESPACE(10,15) )        -
          DATA                        -
          ( NAME(EXAMPLE.KSDS3.DATA)  -
            RECORDS(800) )            -
          INDEX                       -
          ( NAME(EXAMPLE.KSDS3.INDEX) -
            BLOCKS (20) )             -
          CATALOG(AMASTCAT/MCATUPPW)
/*
/&
```

Explanation of the IDCAMS Commands

The DEFINE SPACE command creates space on an FBA volume.

1. The ORIGIN parameter indicates 1600 is the beginning block number.
2. The VOL parameter is required and specifies the volume that is to contain the data space.
3. The BLOCKS parameter specifies the amount of space to be allocated on this volume. An amount of space (the size of the maximum CA, which is device-dependent) will be taken from the 12000 blocks for VSAM itself, because this is the initial definition of space for the volume.

   On a SCSI device, for example, the minimum allocation cannot be smaller than the following inputs:

   ```
   // JOB MINIMUM DEF USERCAT 5 Min-CAs == (5*512)+1 BLOCKS
   // DLBL IJSYSUC,'USERCAT1',,VSAM
   // ASSGN SYS000,601
   // EXEC IDCAMS,SIZE=AUTO
      DEFINE USERCATALOG(      -
           BLOCKS(2561  0)     -
           NAME(USERCAT1)      -
           VOLUME(SCSI01)      )
      LISTCAT ALL
   /*
   /&
   ```

   The absolute minimum space specified for primary allocation is 2561 blocks, which is rounded to 3072.

   For FBA devices, it is recommended not to use the VSAM default settings but rather to explicitly define VSAM catalogs with larger primary and secondary allocation sizes. The default catalog on an FBA volume will have 256 blocks as primary and 128 blocks as secondary allocation. These values will result in a catalog-full condition after defining 512 files (including the catalog itself). The catalog has already reached 16 extents (the maximum for VSAM catalogs) and cannot be extended further.
4. The CLASS parameter specifies that the space is to be classified as class 0 (same as the default).
5. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog.

The DEFINE CLUSTER command defines a key-sequenced file into the class 0 space on the FBA volume.

1. The CLUSTER parameter is required and NAME specifies the cluster defined.
2. The RDPW parameter specifies the read password of this cluster. Because no master password is defined, this password will be propagated up to the master level.
3. The VOL parameter specifies the volume containing this file.
4. The RECORDSIZE parameter specifies the average and maximum record sizes.
5. The KEYS parameter specifies the key length and offset from the beginning of the record.
6. The FREESPACE parameter specifies the percentage of space within CIs and CAs, respectively, that is originally free.
7. The DATA parameter is required if attributes are to be specified for the data component. The NAME parameter specifies the name of the data component. If a name is not specified, a name is generated.

   Note: Due to different rounding values (different value for minimum CA on FBA and SCSI devices), it is not ensured that the same JCLs will run on generic FBA and SCSI devices. It is essential to specify a valid value depending on the device type. On a SCSI device, for example, the following specification is required as a minimum:

   ```
   DEFINE CLUSTER (                          -
           NAME (VSAM.TEST.FILE)             -
           RECORDS (200 100)                 -
           SHAREOPTIONS (4 4)                -
           RECORDSIZE (80 500)               -
           VOLUMES (GLOSC1 )                 -
           NOREUSE                           -
           INDEXED                           -
           FREESPACE (15 7)                  -
   ```

```
        KEYS (44 0)                          -
        NOCOMPRESS                           -
        TO (99366))                          -
        DATA  (NAME (VSAM.TEST.FILE.@D@)     -
        CONTROLINTERVALSIZE(1024))           -
        INDEX (NAME (VSAM.TEST.FILE.@I@))    -
        CATALOG (SCSIUCAT)
```

With a CONTROLINTERVALSIZE of 512 the DEFINE CLUSTER will fail because, with the allocation of 512 (min CA of 512 blocks), VSAM first reserves the value for the Index CI, but there is no space left to allocate the Data CI on a SCSI device. The same specification is acceptable for a generic FBA, however.

8. The RECORDS parameter specifies space suballocation in records (RECORDS or BLOCKS are the only valid parameters for an FBA device).

   Restriction: You can specify a maximum value of 16,777,215 (X'FFFFFF') in the RECORDS parameter. If RECORDS is specified at the cluster level, the space must be large enough to include the index records; otherwise, the file will not hold the expected number of data records.

9. The INDEX parameter is required if attributes are to be specified for the index component. The NAME parameter specifies the name of the index component. If a name is not specified, a name is generated by VSE/VSAM.
10. The BLOCKS parameter specifies space suballocation in blocks (RECORDS or BLOCKS are the only valid parameters for an FBA device).

    Restriction: You can specify a maximum value of 16,777,215 (X'FFFFFF') in the BLOCKS parameter. If BLOCKS is specified at the cluster level, the space must be large enough to include the index records; otherwise, the file will not hold the expected number of data records.

11. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog.

Example 10: Exporting a VSE/VSAM File

This example exports an entry-sequenced file from a user catalog named D27UCAT2 to a labeled tape. Following the successful creation of the portable copy, the temporary export flag is set for the file in D27UCAT2 as well as the inhibit update flag, which will prevent any access other than read access to the file. When the file is imported, the inhibit update flag will be set for the file in the target catalog.

Example 10 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL10
*  EXPORT REQUIRES SYS005 TO BE ASSIGNED TO AN OUTPUT TAPE FILE
// ASSGN   SYS005,cuu
// DLBL    IJSYSUC,'D27UCAT2',,VSAM
// TLBL    RECEIVE,'PORTABLE.TAPE1',,TAPE01,1
// EXEC    IDCAMS,SIZE=AUTO
   EXPORT  EXAMPLE.ESDS1/UCATMRPW               -
           OUTFILE(RECEIVE,                     -
             ENV(PDEV(2400) NOREWIND STDLABEL))  -
           RECORDMODE                           -
           TEMP                                 -
           INHIBITSOURCE                        -
           INHIBITTARGET
/*
/&
```

Explanation of JCL

1. SYS005 must be used in the ASSGN statement for magnetic tape output (OUTFILE).
2. The IJSYSUC DLBL statement describes the job catalog that contains the entry-sequenced file to be exported.
3. The RECEIVE TLBL statement describes the portable file that will contain the exported file.

Explanation of the IDCAMS Command

The EXPORT command causes an entry-sequenced file from a user catalog named D27UCAT2 to be exported to the first file of a labeled tape.

1. The name of the cluster object is required. The master password is required to export a password-protected object.
2. The OUTFILE parameter is required and identifies the portable file. The filename of the TLBL statement for this file must be identical to this name. The PDEV subparameter must be specified for a tape device. The BLKSZ subparameter defaults to 2048. The NOREWIND subparameter is specified because the file is on a multifile volume. The following examples will add files to this tape. The STDLABEL subparameter specifies standard label processing for the tape.
3. The RECORDMODE parameter overrides the default (CIMODE) and causes EXPORT processing using record mode operations.
4. The TEMP parameter causes the temporary export attribute to be set and prevents the file from being deleted.
5. The INHIBITSOURCE parameter causes the inhibit update flag to be in the catalog that contained the entry for the file being exported.
6. The INHIBITTARGET parameter causes the inhibit update flag to be set in the target catalog when the file is imported.

Example 11: Exporting an Alternate Index and Base Cluster

This example exports the alternate index and base cluster from the master catalog. Any paths defined over either object will be exported with their PATHENTRY object. Because the export is permanent, both the base cluster and alternate index will be deleted from the catalog. The alternate index must be exported first as the delete of the base cluster will cause deletion of all objects defined over it.

Example 11 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL11
*  EXPORT REQUIRES SYS005 TO BE ASSIGNED TO AN OUTPUT TAPE FILE
// ASSGN   SYS005,cuu
// TLBL    RECEIVE,'PORTABLE.TAPE2',,TAPE01,1
// EXEC    IDCAMS,SIZE=AUTO
   EXPORT  EXAMPLE.AIX/MCATMRPW                    -
           OUTFILE(RECEIVE,                        -
             ENV(PDEV(2400) BLKSZ(2056)NOREWIND))
/*
// TLBL    RECEIVE,'PORTABLE.TAPE3',,TAPE01,1
// EXEC    IDCAMS,SIZE=AUTO
   EXPORT  EXAMPLE.KSDS2/MCATMRPW                  -
           OUTFILE(RECEIVE,                        -
             ENV(PDEV(2400) BLKSZ(2056)NOREWIND))
/*
// MTC     RUN,SYS005
/&
```

Explanation of JCL

1. SYS005 must be used in the ASSGN statement for magnetic tape output (OUTFILE).
2. The RECEIVE TLBL statements describe the portable files. Note that these are the second and third files on a multifile volume.
3. The MTC statement rewinds and unloads the portable copy tape after the EXPORT commands are completed.

Explanation of the IDCAMS Commands

The first EXPORT command causes an alternate index to be exported from the master catalog.

1. The name of the alternate index being exported is required. A master password is required for the deletion and to allow VSE/VSAM locates against both the alternate index and path to obtain the catalog

information (including passwords) to be exported. The master password of the catalog covers all requirements.

2. The OUTFILE parameter is required and identifies the portable file. The filename of the TLBL statement for this object must be identical to this name. The PDEV subparameter must be specified for a tape device. Normally, the BLKSZ subparameter defaults to 2048; however, for improved performance it is recommended that you specify a value at least 8 bytes larger than the file's data-component CI size. The NOREWIND subparameter is specified because the file is on a multifile volume.
3. CI processing (CIMODE) is effective as the default.
4. PERMANENT export is effective as the default.

The second EXPORT command causes a base cluster to be exported from the master catalog.

1. The name of the base cluster being exported is required. Because the cluster level is not protected, no password would be required for the deletion. However, a password is required for the VSE/VSAM locates against the data and index components to obtain the catalog information (including passwords) to be exported. Because only one password can be supplied, it must be that of the master catalog.
2. The OUTFILE parameter is required and identifies the portable file. The filename of the TLBL statement for this object must be identical to this name. The PDEV subparameter must be specified for a tape device. Normally the BLKSZ subparameter defaults to 2048; however, for improved performance it is recommended that you specify a value at least 8 bytes larger than the file's data-component CI size. The NOREWIND subparameter is specified because the file is on a multifile volume.
3. The RECORDMODE parameter is not specified so processing defaults to CIMODE processing.
4. PERMANENT export is effective as the default.

Example 12: Disconnecting a User Catalog from a Master Catalog

This example disconnects user catalog D27UCAT1 from the master catalog. The user catalog volume need not be mounted; therefore, no DLBL statement to describe it is required.

Example 12 can be viewed as a logical follow on to the previous examples.

```
// JOB     EXAMPL12
// EXEC    IDCAMS,SIZE=AUTO
   EXPORT  D27UCAT1/MCATUPPW -
           DISCONNECT
/*
/&
```

Explanation of the IDCAMS Command

1. The name of the user catalog is required. The update or higher-level password of the master catalog is required to disconnect the user catalog.
2. The DISCONNECT parameter is required to signal the disconnect action.

Example 13: Importing a Base Cluster and Alternate Index

This example will import the base cluster and alternate index exported in Example 11 into the master catalog. The importation causes every component to be newly defined. Because the alternate index cannot be defined until the base cluster has been defined, the base cluster must be imported first. In order to do this, manual positioning of the tape is shown. The importation will cause any paths over the objects which were exported to be redefined.

Example 13 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL13
*  IMPORT REQUIRES SYS004 TO BE ASSIGNED TO AN INPUT TAPE FILE
// ASSGN   SYS004,cuu
*  POSITION TO THE THIRD FILE ON THE TAPE
// MTC     REW,SYS004
// MTC     FSF,SYS004,6
```

```
// TLBL    SOURCE,'PORTABLE.TAPE3',,TAPE01,1
// EXEC    IDCAMS,SIZE=AUTO
   IMPORT  INFILE(SOURCE                          -
                  ENV(PDEV(2400) BLKSZ(2056)))    -
           OBJECTS((EXAMPLE.KSDS2.INDEX           -
                    USECLASS(0 0)))               -
           CAT(AMASTCAT/MCATUPPW)
/*
*  POSITION TO THE SECOND FILE ON THE TAPE
/*
// MTC     REW,SYS004
// MTC     FSF,SYS004,3
// TLBL    SOURCE,'PORTABLE.TAPE2',,TAPE01,1
// EXEC    IDCAMS,SIZE=AUTO
   IMPORT  INFILE(SOURCE                          -
                  ENV(PDEV(2400) BLKSZ(2056)))    -
           CAT(AMASTCAT/MCATMRPW)
/*
// MTC     RUN,SYS004
/&
```

Explanation of JCL

1. SYS004 must be used in the ASSGN statement for magnetic tape input (INFILE).
2. The first MTC command positions the tape at the load point.
3. The second MTC command positions the tape at the third file on the tape (EXAMPLE.KSDS2).
4. The first SOURCE TLBL statement describes the portable files.
5. The third MTC command positions the tape at the load point.
6. The fourth MTC command positions the tape at the second file on the tape (EXAMPLE.AIX).
7. The second SOURCE TLBL statement describes the portable file.

Explanation of the IDCAMS Commands

The first IMPORT command causes the base cluster to be imported into the master catalog. The class of space from which the index component is to be allocated is changed from that contained on the portable file.

1. The INFILE parameter is required and identifies the portable file. The PDEV subparameter must be specified for a tape device. The BLKSZ subparameter must be specified because the IMPORT uses the same value (2056) that the EXPORT used. The filename of the TLBL statement for this object must be identical to this name. NOREWIND and STDLABEL are default values.
2. The OBJECTS parameter identifies the index component of the file being imported. The USECLASS parameter specifies that the index component is to be allocated from class 0 space. This specification overrides the useclass attributes in the portable file.
3. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define into a protected catalog.

The second IMPORT command causes an alternate index to be imported into the master catalog.

1. The INFILE parameter is required and identifies the portable file. The PDEV subparameter must be specified for a tape device. The BLKSZ subparameter must be specified because the IMPORT uses the same value (2056) that the EXPORT used. The filename of the TLBL statement for this object must be identical to this name. NOREWIND and STDLABEL are default values.
2. The CATALOG parameter is required because the master catalog is password-protected. It specifies the name of the master catalog and its update password which is required to define an alternate index over a password-protected base cluster.

Example 14: Importing an Entry-Sequenced File

This example imports the entry-sequenced file exported in Example 10. The exported copy of the file will replace the original copy of the file. IMPORT will find the duplicate name, EXAMPLE.ESDS1, in catalog

D27UCAT2, delete it because the temporary export attribute will have been set for EXAMPLE.ESDS1, then redefine it using the catalog information from the portable file.

Example 14 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL14
*  IMPORT REQUIRES SYS004 TO BE ASSIGNED TO AN INPUT TAPE FILE
// ASSGN   SYS004,cuu
// MTC     REW,SYS004
// TLBL    SOURCE,'PORTABLE.TAPE1',,TAPE01,1
// EXEC    IDCAMS,SIZE=AUTO
   IMPORT  INFILE(SOURCE                          -
                  ENV(PDEV(2400)))                -
           CATALOG(D27UCAT2/UCATMRPW)
/*
// MTC     RUN,SYS004
/&
```

Explanation of JCL

1. SYS004 must be used in the ASSGN statement for magnetic tape input (INFILE).
2. The MTC command positions the tape at the first file (EXAMPLE.ESDS1).
3. The SOURCE TLBL statement describes the portable file. Note that the second file on the volume is specified.

Explanation of the IDCAMS Command

The IMPORT command causes an entry-sequenced file to be imported from the first file of a labeled tape into a user catalog named D27UCAT2.

1. The INFILE parameter is required and identifies the portable file. The filename of the TLBL statement for this file must be identical to this name. The PDEV subparameter must be specified for a tape device. The BLKSZ subparameter need not be specified because IMPORT will use the same default (2048) that EXPORT used. NOREWIND and STDLABEL are default values.
2. The CATALOG parameter is required because the target catalog is not the default catalog and is password-protected. It specifies the name of the user catalog. The update password of the catalog permits IMPORT to delete the duplicate entry as well as to redefine the imported file.

Example 15: Connecting a User Catalog to the Master Catalog

This example connects user catalog D27UCAT1 to the master catalog. The volume containing the user catalog need not be mounted. Example 15 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL15
// EXEC    IDCAMS,SIZE=AUTO
   IMPORT  CONNECT               -
           OBJECTS               -
           ( (D27UCAT1           -
             VOLUME(VSER02)      -
             DEVICETYPE(3380))   -
           )                     -
           CATALOG(AMASTCAT/MCATUPPW)
/*
/&
```

Explanation of the IDCAMS Command

1. The name of the user catalog is required.
2. The VOLUME subparameter is required to identify the volume containing the user catalog.
3. The DEVICETYPE subparameter is required to identify the device type of the volume containing the user catalog.
4. The CONNECT parameter is required to signal the connect action.

5. The CATALOG parameter is required because the master catalog's update or higher-level password is required if the master catalog is password-protected.

Example 16: Using REPRO to Unload a User Catalog to Tape

This example shows how a VSE/VSAM user catalog can be unloaded to tape using the catalog unload/reload function of the REPRO command.

Example 16 can be viewed as a logical follow-on to the previous examples.

```
// JOB    EXAMPL16
*  REPRO REQUIRES SYS005 TO BE ASSIGNED TO AN OUTPUT TAPE FILE
// ASSGN  SYS005,cuu
// MTC    REW,SYS005
// DLBL   IJSYSUC,'D27UCAT2',,VSAM
// TLBL   CATOUT,'PORTABLE.TAPE1',,TAPE01,1
// EXEC   IDCAMS,SIZE=AUTO
   REPRO  INFILE(IJSYSUC/UCATMRPW)   -
          OUTFILE                    -
          ( CATOUT                   -
            ENVIRONMENT              -
            (PDEV(2400)              -
             RECFM(VARBLK)           -
             REW                     -
             BLKSZ(5164)             -
             RECSZ(516))             -
          )
/*
/&
```

Explanation of JCL

1. SYS005 must be used in the ASSGN statement for magnetic tape output (OUTFILE).
2. The MTC command positions the tape at the load point.
3. The IJSYSUC DLBL statement is required and describes the catalog as the job catalog for the job (and as a VSE/VSAM file to be opened and used by REPRO as the source for the unload operation).
4. The CATOUT TLBL statement describes a tape file which is to hold the unloaded copy of the catalog.

Explanation of the IDCAMS Command

The REPRO command causes a VSE/VSAM user catalog to be unloaded to tape.

1. The INFILE parameter is required and identifies the job catalog as an input VSE/VSAM file. The master password allows opening the catalog as a file.
2. The OUTFILE parameter is required and describes the tape file which is to hold the unloaded copy of the catalog. The filename of the TLBL statement for this file must be identical to this name. The PDEV subparameter must be specified for a tape device. The RECFM subparameter must be specified to designate a variable blocked format.
3. The REW parameter indicates rewind the tape at completion.
4. The BLKSZ subparameter, in the case of a catalog file, must be a multiple of 516 plus 4. The RECSZ subparameter must be specified because the default is block size minus 4 and 516 is the only acceptable value (length of the catalog record +4).

Example 17: Using REPRO to Reload a User Catalog

This example shows how a VSE/VSAM user catalog can be reloaded from the backup copy created in Example 16 using the catalog unload/reload function of the REPRO command.

Example 17 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL17
*  REPRO REQUIRES SYS004 TO BE ASSIGNED TO AN INPUT TAPE FILE
// ASSGN   SYS004,cuu
// MTC     REW,SYS004
```

```
// DLBL    IJSYSUC,'D27UCAT2',,VSAM
// TLBL    CATIN,'PORTABLE.TAPE1',,TAPE01,1
// EXEC    IDCAMS,SIZE=AUTO
   REPRO   INFILE            -
           ( CATIN           -
             ENVIRONMENT     -
             (PDEV(2400)     -
              RECFM(VARBLK)  -
              BLKSZ(5164)    -
              RECSZ(516))    -
           )                 -
          OUTFILE(IJSYSUC/UCATMRPW)
/*
/&
```

Explanation of JCL

1. SYS004 must be used in the ASSGN statement for magnetic tape input (INFILE).
2. The MTC command positions the tape at the load point.
3. The IJSYSUC DLBL statement is required and describes the user catalog to be reloaded as the job catalog (and as a VSE/VSAM file to be opened and used by REPRO as the target for the reload operation).
4. The CATIN TLBL statement describes the backup copy of the user catalog as a tape file.

Explanation of the IDCAMS Command

The REPRO command causes a VSE/VSAM user catalog to be reloaded from the backup copy created in Example 16.

1. The INFILE parameter is required and identifies the tape file which holds the backup copy of the user catalog. The filename of the TLBL statement for this file must be identical to this name. The PDEV subparameter must be specified for a tape device. The RECFM subparameter must be specified to designate a variable blocked format. The BLKSZ subparameter must be specified to ensure that it is the same as that specified when the catalog was unloaded. The RECSZ subparameter must be specified because the default is block size minus 4 and 516 is the only acceptable value (length of the catalog record +4).
2. The OUTFILE parameter is required and identifies the job catalog as a VSE/VSAM file to be opened. The master password allows opening the catalog as a file.

Example 18: Deleting Entries in a User Catalog and the User Catalog Itself

This example deletes entries in the VSE/VSAM user catalog and deletes the user catalog, D27UCAT2. The result of this job and that shown by Example 27 is to remove all VSE/VSAM files and catalogs defined in this set of examples.

Example 21 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL21
// EXEC    IDCAMS,SIZE=AUTO
   /* DELETE THE NONVSAM FILE EXAMPLE.NONVSAM1 */
   DELETE  EXAMPLE.NONVSAM1               -
           SCRATCH                        -
           CATALOG(D27UCAT1)
   /* DELETE VSE/VSAM FILE CATALOGUED IN */
   /* USER CATALOG D27UCAT2 */
   DELETE  (EXAMPLE.ESDS1/ESD1PWMR)       -
           PURGE                          -
           CLUSTER                        -
           CATALOG(D27UCAT2)
   /* DELETE VSE/VSAM FILES IN MASTER CATALOG */
   DELETE  (EXAMPLE.KSDS1/KSD1PSWD)       -
           PURGE                          -
           CLUSTER                        -
   /* DELETE THE VSE/VSAM USER CATALOG D27UCAT2 */
   DELETE  D27UCAT2/UCATMRPW              -
           USERCATALOG                    -
           PURGE
```

```
/*
/&
```

Explanation of the IDCAMS Commands

The first DELETE command deletes the nonVSAM file named EXAMPLE.NONVSAM1.

1. The name of the nonVSAM file is required.
2. The SCRATCH parameter specifies that the VTOC entry of the object being deleted is to be removed.
3. The NONVSAM parameter insures that the entry being deleted is a nonVSAM file.
4. The CATALOG parameter is required in this example. It identifies the catalog containing the entry to be deleted. A password is not required to delete a nonVSAM entry.

The second DELETE command deletes the VSE/VSAM file cataloged in the user catalog D27UCAT2.

1. The name of the VSE/VSAM entry-sequenced file is required. Its master password is supplied.

   Note: The password supplied is the value specified in the ALTER password in example 7.

2. The PURGE parameter causes the entries to be deleted without regard for the retention period.
3. The CLUSTER parameter insures that the catalog object being deleted is a VSE/VSAM file.
4. The CATALOG parameter is required in this example. It identifies the catalog containing the entry to be deleted. Its password is not required because the file was specified with its master password.

The third DELETE command deletes the VSE/VSAM file cataloged in the master catalog.

1. The name of the VSE/VSAM key-sequenced file and its master password are required.
2. The PURGE parameter causes the entries to be deleted without regard for the retention period.
3. The CLUSTER parameter ensures that the catalog object being deleted is a VSE/VSAM file.

   Note: The default catalog is the master catalog.

The fourth DELETE command deletes the VSE/VSAM user catalog D27UCAT2.

1. The name of the user catalog and its master password are required.
2. The USERCATALOG parameter is required to specify that the object being deleted is a user catalog.
3. The PURGE parameter causes the entry to be deleted without regard for the retention period.

Example 19: Defining an Entry Sequence Default Model

This example creates a default model for a VSE/VSAM entry-sequence file.

Example 22 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL22
// EXEC    IDCAMS,SIZE=AUTO
   DEFINE  CLUSTER                      -
           ( NAME(DEFAULT.MODEL.ESDS)   -
             NOALLOCATION               -
             CYLINDERS(1 1)             -
             RECORDSIZE(2500 3000)      -
             SPANNED                    -
             VOLUMES(VSER01)            -
             NONINDEXED )               -
           CATALOG(AMASTCAT/MCATMRPW)
/*
/&
```

Explanation of the IDCAMS Command

1. The cluster parameter is required and NAME specifies the name of the cluster being defined. For a default model, the name must begin with DEFAULT.MODEL.
2. The NOALLOCATION parameter indicates that no space allocation is to take place for the DEFINE.

3. A space parameter, in this example CYLINDERS, is required because the MODEL parameter is not specified. One cylinder is specified for both the primary and secondary allocations.
4. The RECORDSIZE parameter specifies the average record size as 2500 bytes and the maximum size as 3000 bytes.
5. The SPANNED parameter indicates that this file can contain records that span more than one CI.
6. The VOLUMES parameter specifies that the cluster will reside on VSER01. This volume also serves as a default volume for entry-sequenced files defined without a VOLUME parameter.
7. The NONINDEXED parameter specifies that the cluster is for an entry-sequenced file.
8. The CATALOG parameter specifies the name and password of the catalog in which the cluster will be defined.

Example 20: Defining a Dynamic File

This example defines a VSE/VSAM entry-sequenced file using the default volume capability and the REUSE and NOALLOCATION (dynamic file) options.

Example 23 can be viewed as a logical follow-on to the previous examples.

```
// JOB      EXAMPL23
// EXEC     IDCAMS,SIZE=AUTO
   DEFINE   CLUSTER                 -
            ( NAME(EXAMPLE.ESDS3)   -
              RECORDSIZE(100 200)   -
              RECORDS(1000 200)     -
              NONINDEXED            -
              REUSE                 -
              NOALLOCATION )        -
            CATALOG(AMASTCAT/MCATMRPW)
/*
/&
```

Explanation of the IDCAMS Command

1. The CLUSTER parameter is required and NAME specifies the name of the cluster being defined.
2. The RECORDSIZE parameter specifies the average and maximum record sizes.
3. The RECORDS parameter specifies the amount of space to be allocated to the file when the file is accessed.
4. The NONINDEXED parameter is required to override the default (INDEXED).
5. The REUSE and NOALLOCATION parameters indicate that the file has the dynamic file capability.
6. The default volumes capability is effective as the default because VOLUMES was not specified. (In this case, a default model for an entry-sequenced file must exist in the catalog.)
7. The CATALOG parameter specifies the name and password of the catalog in which the cluster will be defined.

Example 21: Accessing a Dynamic File

This example accesses the reusable VSE/VSAM entry-sequenced file defined in Example 23 using the DLBL statement's DISP= operand. The file is allocated at OPEN, loaded and kept (passed) at CLOSE in the first step of the job. The second step accesses the passed data and then deallocates (that is, sets the file back to the NOALLOCATION state) the file at CLOSE.

Example 24 can be viewed as a logical follow-on to the previous examples.

```
// JOB   EXAMPL24
// DLBL  FILE1,'EXAMPLE.ESDS3',,VSAM,DISP=(NEW,KEEP)
// EXEC  PROG1,SIZE=AUTO
// DLBL  FILE2,'EXAMPLE.ESDS3',,VSAM,DISP=(OLD,DELETE,KEEP)
// EXEC  PROG2,SIZE=AUTO
/*
/&
```

Explanation of JCL

1. The first DLBL statement indicates that the file should have space allocated when it is opened. The disposition KEEP indicates that this space should not be freed when the file is closed.
2. The first EXEC statement implies that PROG1 opens and closes the file.
3. The second DLBL statement indicates that the file already has space allocated to it. The disposition DELETE indicates that this space should be deleted - but only when the file is *normally* closed. If the job is ended *abnormally*, the disposition KEEP indicates that the file is to be kept, so that you can rerun the job without reloading the file.
4. The second EXEC statement implies that PROG2 opens and closes the file.

Example 22: Defining a Partition-Independent File

This example creates a VSE/VSAM entry-sequence file. The file name will be "PART.INDEP.xx" where xx is the identifier of the partition in which the job is executed, that is, BG, F1, F2, etc. As specified in this example, the command does not provide all the parameters necessary for successful file definition. These parameters will be filled in by VSE/VSAM according to the values specified in the default model (DEFAULT.MODEL.ESDS) specified in Example 24.

Example 25 can be viewed as a logical follow-on to the previous example.

```
// JOB     EXAMPL25
// EXEC    IDCAMS,SIZE=AUTO)
   DEFINE  CLUSTER                    -
           (NAME(%PART.INDEP)         -
            NONINDEXED)               -
           CATALOG(AMASTCAT/MCATUPPW)
/*
/&
```

Explanation of the IDCAMS Command

1. The CLUSTER parameter is required and NAME specifies the name of the cluster being defined. The % on the name indicates a partition-independent name.
2. The NONINDEXED parameter specifies that the cluster is for an entry-sequenced file.
3. The CATALOG parameter specifies the name and password of the catalog in which the cluster will be defined.

Example 23: Syntax Checking a Command Stream

This example shows how to syntax check IDCAMS commands. As long as you specify syntax checking, the commands are not executed and data is not modified.

Example 26 can be viewed as a logical follow-on to the previous examples.

```
// JOB    EXAMPL26
/* COMMANDS AND MACROS BOOK EXAMPLE
// EXEC   IDCAMS,SIZE=AUTO,PARM='SYNCHK'
   DELEYE (TEST1) CL                      See Note 1
 DEFINE CLUATER         -                 See Note 1
        (NAME(TEST1)    -
        NUMBERED        -
        INDEXED)
/*
/&
// JOB    EXAMPL26 2ND TRY
// EXEC  IDCAMS,SIZE=AUTO,PARM='SYNCHK'    See Note 2
 DELETE  (TEST1) CL
 DEFINE  CLUSTER        -
         (NAME(TEST1)   -
         NUMBERED       -
         INDEXED)
/*
/&
// JOB   EXAMPL26 3RD TRY
// EXEC  IDCAMS,SIZE=AUTO,PARM='SYNCHK'    See Note 3
```

```
 DELETE  (TEST) CL
 DEFINE  CLUSTER        -
         (NAME(TEST1)   -
         INDEXED)
/*
/&
```

Explanation of JCL

The PARM='SYNCHK' on the EXEC statement indicates that only syntax checking is done and the commands are not to be executed.

Explanation of Commands

1. Note 1. The first EXEC IDCAMS statement has errors in both IDCAMS commands:
   - DELETE is misspelled as DELEYE.
   - CLUSTER is misspelled as CLUATER in the DEFINE command.
2. Note 2. The second EXEC IDCAMS statement:
   - Shows both the previous errors corrected. That is, the DELETE command is now correct.
   - The DEFINE command still has inconsistent parameters (NUMBERED and INDEXED).
3. Note 3. The third EXEC IDCAMS statement shows the DEFINE error (inconsistent parameters) corrected.

Because both IDCAMS commands are correct, the example is free of syntax errors.

Example 24: Deleting Entries in the Master Catalog and the Master Catalog Itself

This example deletes the second user catalog, the entries in the master catalog, and, finally, the master catalog itself.

Example 27 can be viewed as a logical follow-on to the previous examples.

```
// JOB     EXAMPL27
// EXEC    IDCAMS,SIZE=AUTO
   /* DELETE THE VSE/VSAM USER CATALOG D27UCAT1  */
   DELETE  D27UCAT1/UCATMRPW                    -
           USERCATALOG                          -
           PURGE
   /* DELETE VSE/VSAM ENTRY-SEQUENCED FILE NAMED */
   /* EXAMPLE.ESDS2 FROM THE MASTER CATALOG    */
   DELETE  EXAMPLE.ESDS2/ESD2MRPW               -
           PURGE                                -
           ERASE
   /* DELETE VSE/VSAM KEY-SEQUENCED FILE NAMED   */
   /* EXAMPLE.KSDS2 FROM THE MASTER CATALOG    */
   DELETE  EXAMPLE.KSDS2                        -
           CATALOG(AMASTCAT/MCATMRPW)
   /* DELETE VSE/VSAM RELATIVE-RECORD FILE NAMED */
   /* EXAMPLE.RRDS1 FROM THE MASTER CATALOG    */
   DELETE  EXAMPLE.RRDS1                        -
           CLUSTER
   /* DELETE THE PARTITION INDEPENDENT FILE    */
   DELETE  %PART.INDEP                          -
           CLUSTER
   /* DELETE EXAMPLE.ESDS3                     */
   DELETE  EXAMPLE.ESDS3                        -
           CLUSTER
   /* DELETE THE DEFAULT MODEL                 */
   DELETE DEFAULT.MODEL.ESDS                    -
       CLUSTER
   /* DELETE VSE/VSAM SPACE FROM VOLUME VSER04   */
   DELETE  VSER04                               -
           SPACE                                -
           CATALOG(AMASTCAT/MCATUPPW)
   /* DELETE VSE/VSAM SPACE FROM VOLUME VSER03   */
   DELETE  VSER03                               -
           SPACE                                -
```

```
            CATALOG(AMASTCAT/MCATMRPW)
    /* DELETE THE VSE/VSAM MASTER CATALOG         */
    DELETE  AMASTCAT/MCATMRPW                    -
            MASTERCATALOG                        -
            PURGE
/*
/&
```

Explanation of Commands

The first DELETE command deletes the VSE/VSAM user catalog D27UCAT1.

1. The name of the user catalog and its master password are required.
2. The USERCATALOG parameter is required to specify that the object being deleted is a user catalog.
3. The PURGE parameter causes the entry to be deleted without regard for the retention period.

The second DELETE command deletes the VSE/VSAM entry-sequenced file EXAMPLE.ESDS2 from the master catalog.

1. The name of the VSE/VSAM file is required. Its master password is supplied.
2. The PURGE parameter causes the entry to be deleted without regard for the retention period.
3. The ERASE parameter causes the data component to be overwritten with binary zeros. As a result, sensitive data is not left as a residue after the file is deleted.

The third DELETE command deletes the VSE/VSAM key-sequenced file EXAMPLE.KSDS2 from the master catalog. This command will cause the deletion of all objects defined over this base cluster, that is, the alternate index named EXAMPLE.AIX, which, in turn, will cause the deletion of paths over that object, that is, EXAMPLE.PATH.

1. The name of the VSE/VSAM file being deleted is required.
2. The CATALOG parameter identifies the catalog containing the file. The master password of the catalog covers the password requirements for all objects being deleted.

The fourth DELETE command deletes the VSE/VSAM relative-record file EXAMPLE.RRDS1 from the master catalog.

1. The name of the VSE/VSAM file being deleted is required. No password is required because the file is not password-protected.
2. The CLUSTER parameter insures that the catalog object being deleted is a VSE/VSAM file.

The fifth DELETE command deletes the partition independent work file PART.INDEP.BG.

1. The % name indicates that the partition identifier is to be appended prior to issuing the delete.
2. The CLUSTER parameter is needed to show that this is a VSE/VSAM cluster.

The sixth DELETE command deletes the entry-sequenced data set.

1. The name of the file is required. No password is required because the file is not password-protected.
2. The CLUSTER parameter is needed to show that this is a VSE/VSAM cluster.

The seventh DELETE command deletes the default model.

1. The name of the model is required. No password is required because the file is not password-protected.
2. The CLUSTER parameter is required to indicate that this is a VSE/VSAM cluster.

The eighth DELETE command deletes the VSE/VSAM data space from volume VSER04. Because this volume no longer has VSE/VSAM files in the master catalog, it will no longer be available to the master catalog. If other VSE/VSAM files still required this volume, the deletion would free only space belonging to the master catalog that was defined separately and is now completely empty.

1. The volume serial number of the volume is required.

2. The SPACE parameter is required to specify that the catalog object being deleted is a VSE/VSAM data space.
3. The CATALOG parameter identifies the catalog containing the VSE/VSAM data space entry. The update or higher password of the catalog is required to delete a volume or space.

The ninth DELETE command deletes the VSE/VSAM data spaces from volume VSER03. Because this volume no longer has VSE/VSAM files, it will cease to belong to the master catalog. If other VSE/VSAM files still required this volume, the deletion would free only space that was separately defined and is now completely empty.

1. The volume serial number of the volume is required.
2. The SPACE parameter is required to specify that the catalog object being deleted is a VSE/VSAM data space.
3. The CATALOG parameter identifies the catalog containing the VSE/VSAM data space entry. The update or higher password of the catalog is required to delete a volume or space.

The last DELETE command deletes the VSE/VSAM master catalog itself.

1. The name of the master catalog and its master password are required.
2. The MASTERCATALOG parameter is required to specify that the object being deleted is a master catalog.
3. The PURGE parameter causes the catalog to be deleted without regard for the retention period.

Example 25: IDCAMS PRINT That Allows Printing in Upper and Lower Case

For printing lower case or foreign language special characters a user-provided translate table is required for IDCAM PRINT. It can be provided via the PARM modal command. Refer to “PARM” on page 205.

The GRAPHICS TABLE(mname) parameter of the PARM command specifies the name of a module (accessible through VSE/VSAM CDLOAD) in the sublibrary that contains a 256-byte user-provided translate table. This table defines the graphic characters for each of the possible 256 bit patterns. Any character to be printed is translated to the bit pattern found in such a table at the position corresponding to its numeric value (0-255).

The following module allows printing in upper and lower case:

```
* $$ JOB JNM=VSAMTRT,DISP=D,CLASS=0
// JOB VSAMTRT
// OPTION CATAL
// LIBDEF PHASE,CATALOG=IJSYSRS.SYSLIB
   PHASE VSAMTRT,*
// EXEC ASSEMBLY
VSAMTRT CSECT
        DC    CL16'               '   hex 0X
        DC    CL16'               '   hex 1X
        DC    CL16'               '   hex 2X
        DC    CL16'               '   hex 3X
        DC    CL16'          .<(+|'   hex 4X
        DC    CL16'&&         $*); '   hex 5X
        DC    CL16'-/        ,%_>?'   hex 6X
        DC    CL16'         :#@''="'   hex 7X
        DC    CL16' abcdefghi     '   hex 8X
        DC    CL16' jklmnopqr     '   hex 9X
        DC    CL16'  stuvwxyz     '   hex AX
        DC    CL16'               '   hex BX
        DC    CL16' ABCDEFGHI     '   hex CX
        DC    CL16' JKLMNOPQR     '   hex DX
        DC    CL16'  STUVWXYZ     '   hex EX
        DC    CL16'0123456789     '   hex FX
        END
/*
// EXEC LNKEDT
/&
* $$ EOJ
```

The following JCL demonstrates how to use the PRINT command with this PARM parameter:

```
* $$ JOB JNM=FILEPRT,DISP=D,CLASS=0
// JOB FILEPRT
// DLBL UCAT,'EXAMPLE.USER.CATALOG',,VSAM
// DLBL KSDS,'SAMPLE.FILE',,VSAM,CAT=UCAT
// EXEC IDCAMS,SIZE=AUTO
   PARM GRAPHICS(TABLE(VSAMTRT))
   PRINT INFILE (KSDS) DUMP
/*
/&
* $$ EOJ
```

Examples of Functions for BACKUP

Alternate Tape Support for Backup

Figure 9 on page 240 shows alternate tape support for Backup.

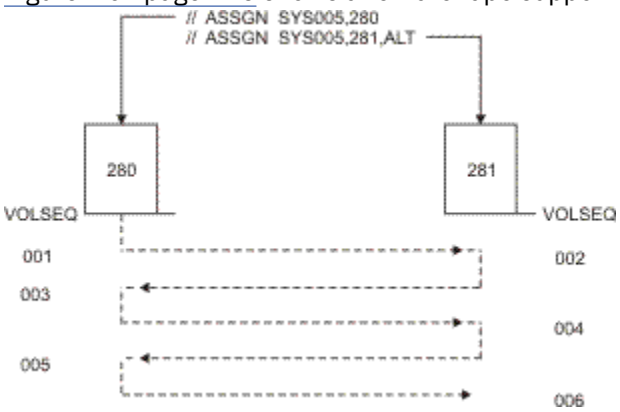


Figure 9. Alternate Tape Support for Backup

Backup Cross-Reference Listings

Backup cross-reference listings are produced by BACKUP and RESTORE commands. For general description of these listings, refer to “Backup Cross-Reference Listings” on page 51.

The following are examples of listings as provided for backup files residing on tape and on disk; for explanations, refer to “Information Concerning Backup Cross-Reference Listings” on page 242.

Listings: Tape Resident Backup Files

Figure 10 on page 240 shows a *volume cross-reference* listing. The entries are sorted by volume sequence numbers.

```
BACKUP VOLUME CROSS-REFERENCE LISTING (BVCR)
VOLSEQ   VOLSER   OBJECT NAME                OBJECT TYPE  SEGMENT TYPE

001      TP6722   HJO.TEST.X1 ............... KSDS         FIRST
002      TP5732   HJO.TEST.X1 ............... KSDS         INTERMEDIATE
003      TP5834   HJO.TEST.X1 ...............KSDS         LAST
                  HJO.TEST.X8 ...............RRDS         ONLY
                  HJO.TEST.X4 ...............ESDS         ONLY
                  HJO.TEST.X3 ...............ESDS         FIRST
004      TP4910   HJO.TEST.X3 ...............ESDS         LAST
                  DEFAULT.MODEL.ESDS.........ESDS         EMPTY
                  DEFAULT.MODEL.ESDS.SAM.....SAM ESDS     EMPTY
                  SAM.WORK.FILE1.............SAM ESDS     EMPTY
                  HJO.TEST.X6 ...............KSDS   CMP   ONLY
                  HJO.TEST.X2 ...............ESDS         ONLY
                  HJO.TEST.X7 ...............ESDS         ONLY
                  HJO.TEST.X5 ...............RRDS         ONLY
```

Figure 10. Backup Volume Cross-Reference Listing (Tape Resident Backup File)

Figure 11 on page 241 shows an *object cross-reference* listing. The entries are sorted by object names.

```
BACKUP OBJECT CROSS-REFERENCE LISTING (BOCR)
OBJECT NAME                  OBJECT TYPE   VOLSEQ  VOLSER   SEGMENT TYPE

DEFAULT.MODEL.ESDS ..........ESDS          004     TP4910   EMPTY
DEFAULT.MODEL.ESDS.SAM ......SAM ESDS      004     TP4910   EMPTY
HJO.TEST.X1 .................KSDS          001     TP6722   FIRST
                                           002     TP5732   INTERMEDIATE
                                           003     TP5834   LAST
HJO.TEST.X2 .................ESDS          004     TP4910   ONLY
HJO.TEST.X3 .................ESDS          003     TP5834   FIRST
                                           004     TP4910   LAST
HJO.TEST.X4 .................ESDS          003     TP5834   ONLY
HJO.TEST.X5 .................RRDS          004     TP4910   ONLY
HJO.TEST.X6 .................KSDS          004     TP4910   ONLY
HJO.TEST.X7 .................ESDS          004     TP4910   ONLY
HJO.TEST.X8 .................RRDS          003     TP5834   ONLY
SAM.WORK.FILE1 ..............SAM ESDS      004     TP4910   EMPTY
```

Figure 11. Backup Object Cross-Reference Listing (Tape Resident Backup File)

Listings: Disk Resident Backup Files

Figure 12 on page 241 shows an *extent cross-reference* listing. The entries are sorted by extent sequence numbers.

```
BACKUP EXTENT CROSS-REFERENCE LISTING (BECR)
EXTSEQ   VOLSER   OBJECT NAME                 OBJECT TYPE  SEGMENT TYPE

001      DSK001   HJO.TEST.X1 ............... KSDS         ONLY
                  HJO.TEST.X8 ............... RRDS         ONLY
                  HJO.TEST.X4 ............... ESDS         FIRST

002      DSK001   HJO.TEST.X4 ............... ESDS         LAST
                  HJO.TEST.X3 ............... ESDS         ONLY
                  DEFAULT.MODEL.ESDS......... ESDS         EMPTY
                  DEFAULT.MODEL.ESDS.SAM..... SAM ESDS     EMPTY

003      DSK002   SAM.WORK.FILE1............. SAM ESDS     EMPTY
                  HJO.TEST.X6 ............... KSDS   CMP   ONLY
                  HJO.TEST.X2 ............... ESDS         ONLY
                  HJO.TEST.X7 ............... ESDS         ONLY
                  HJO.TEST.X5 ............... RRDS         ONLY
```

Figure 12. Backup Extent Cross-Reference Listing (Disk Resident Backup File)

Figure 13 on page 241 shows an *object cross-reference* listing. The entries are sorted by object names.

```
BACKUP OBJECT CROSS-REFERENCE LISTING (BOCR)
OBJECT NAME                  OBJECT TYPE   EXTSEQ  VOLSER   SEGMENT TYPE

DEFAULT.MODEL.ESDS ..........ESDS          002     DSK001   EMPTY
DEFAULT.MODEL.ESDS.SAM ......SAM ESDS      002     DSK001   EMPTY
HJO.TEST.X1 .................KSDS          001     DSK001   ONLY
HJO.TEST.X2 .................ESDS          003     DSK002   ONLY
HJO.TEST.X3 .................ESDS          002     DSK001   ONLY
HJO.TEST.X4 .................ESDS          001     DSK001   FIRST
                                           002     DSK001   LAST
HJO.TEST.X5 .................RRDS          003     DSK002   ONLY
HJO.TEST.X6 .................KSDS          003     DSK002   ONLY
HJO.TEST.X7 .................ESDS          003     DSK002   ONLY
HJO.TEST.X8 .................RRDS          001     DSK001   ONLY
SAM.WORK.FILE1 ..............SAM ESDS      003     DSK002   EMPTY
```

Figure 13. Backup Object Cross-Reference Listing (Disk Resident Backup File)

Figure 14 on page 242 shows an *extent list*. This list provides the following information:

- Volume serial numbers of the disk volumes where the extents are located.
- Limits of the individual disk extents occupied by the backup file.
- Disk address of the backup file record that was written last.

```
BACKUP EXTENT LIST
EXTENT      VOLSER      LOW LIMIT   HIGH LIMIT

001         DSK001      00140000    00140005
002         DSK001      00200005    00210000
003         DSK002      00010000    0005000A

DISK ADDRESS LAST USED : 0002000902
```

Figure 14. Backup Extent List (Disk Resident Backup File)

Information Concerning Backup Cross-Reference Listings

VOLSEQ is the sequence number in which the BACKUP command processed the individual tape volumes.

OBJECT TYPE describes the data set organization of the backed up cluster. The suffix CMP identifies the cluster as a COMPRESSED file. Compressed files can be restored only to systems with VSE/VSAM data compression support (VSE/VSAMVersion 2 and later).

EXTSEQ is the sequence number for the disk sequence. It indicates in which sequence the individual disk extents are occupied by the BACKUP command.

VOLSER is the volume serial number of the appropriate tape or disk volume. For unlabeled tapes, **NONE** is printed under this heading.

SEGMENT TYPE indicates which part of the VSE/VSAM object is contained on the referenced volume or within the corresponding disk extent:

ONLY - the indicated tape volume or disk extent contains the complete VSE/VSAM object.
FIRST - the indicated tape volume or disk extent contains the first part of the VSE/VSAM object.
INTERMEDIATE - the indicated tape volume or disk extent contains an intermediate part of the VSE/VSAM object.
LAST - the indicated tape volume or disk extent contains the last part of the VSE/VSAM volume.
EMPTY - the indicated object was found, but only its catalog description was placed in the backup file because no data is present.

LOW LIMIT is the low limit address for the individual disk extent; that is:

- For CKD or ECKD disks, a disk address in the form ccchhhh
- For FBA disks, a physical block number.

HIGH LIMIT is the high limit address for the individual disk extent; that is:

- For CKD or ECKD disks, a disk address in the form ccchhhh
- For FBA disks, a physical block number.

DISK ADDRESS LAST USED is the disk address of the last backup file record written; that is:

- For CKD or ECKD disks, a disk address in the form ccchhhh
- For FBA disks, a physical block number.

Generic Names

If you wish to back up several related (by name) VSE/VSAM objects at a time, you can specify a *generic name* in the BACKUP command instead of the individual entrynames of all the related objects if the names have the same leading name segment(s). A generic name as supported by VSE/VSAM Backup/Restore is an entryname that contains an asterisk (*) as the *last* name segment.

A generic name represents a group of entrynames, namely the set of all entrynames (including empty objects) contained in the default catalog that start with the name segments preceding the asterisk in the generic name. Do not specify a partial name segment. Figure 15 on page 243 shows the effect of using a generic name for backup. Notice that the asterisk can represent more than one name segment. HJO. must be specified completely (including the period). HJ* is not valid.

There is no formal restriction on the number of VSE/VSAM objects that can be backed up with a generic name. There are, however, practical restrictions because every VSE/VSAM object requires a small amount of virtual storage.

Using a generic name provides an easy way to back up *all* VSE/VSAM objects of a catalog. To achieve this, simply specify "*" in the BACKUP command. Do not specify an asterisk for an object whose name is only one segment because *all* objects will be backed up.

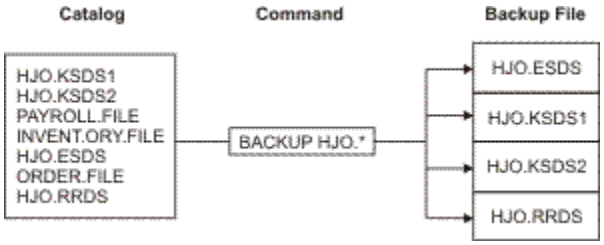


Figure 15. Generic Backup

Excluding Objects from Backup

If not all objects represented by a generic name are to be backed up, use the EXCLUDE parameter. EXCLUDE takes precedence over the entryname list; none of the objects named in the EXCLUDE parameter (individually or by generic name) are backed up, even if they are specified in the entryname list. This is illustrated in Figure 16 on page 243.

Note: If no object is found for a specified entryname (whether or not the object was excluded), a warning message is printed so that you can remove the superfluous entryname the next time you run the backup or restoration job.

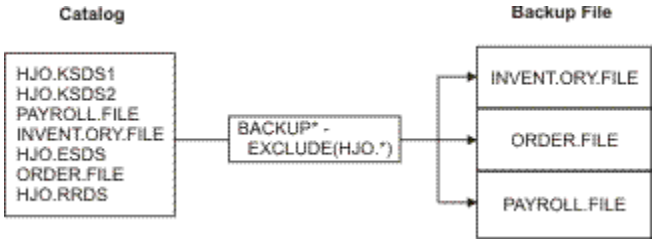


Figure 16. Exclusion from Backup

Excluding Associated Objects

When an object is a candidate for backup, all its associated objects are automatically backed up with it. Such automatic backup can be *suppressed* by explicitly naming the objects to be suppressed in the EXCLUDE parameter.

Figure 17 on page 244 shows what happens when objects are automatically backed up.
Figure 18 on page 244, then, shows what goes to the backup file when you suppress the object *VSAM.KSDS.AIX2*.

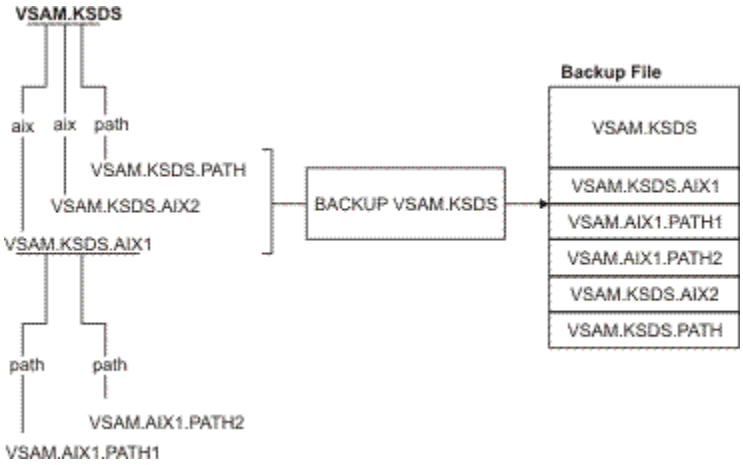


Figure 17. Automatic Backup of Alternate Indexes and Paths

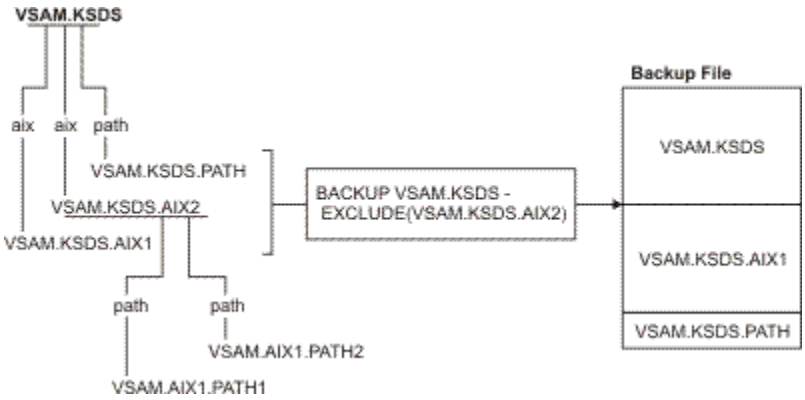


Figure 18. Exclusion from Automatic Backup

Buffer Allocation for Backup

VSE/VSAM Backup/Restore uses buffers for reading and writing both VSE/VSAM file and backup file information. The size and number of these buffers determine performance.

The size of the buffers is specified in the BLOCKSIZE parameter of the BACKUP command. The size determines the size of the data blocks of the backup file and influences the streaming mode (if available and used) of a tape device.

The number of buffers available to the backup function is specified in the BUFFERS parameter of the BACKUP command. This parameter is intended to assist streaming in a multiprogramming environment.

Considerations for Backup Files on Tape

- The trailing reflective marker on any backup volume must be sufficiently far from the end of the tape reel to accommodate as many buffers as specified in the BUFFERS parameter, plus some small control records.
- Whether or not the buffers should be fixed in real storage can be specified via the UPSI job control statement.
- VSE/VSAM Backup/Restore optimizes resource allocation according to the device used. Under normal circumstances, you need not specify the BLOCKSIZE and BUFFERS parameters. However, you may use the parameters to further tune your system operations.

Backup Job Examples

For the subsequent examples, it is assumed that the following catalogs and objects exist:

Master Catalog

Name: VSAM.MASTER.CATALOG

Objects:

KSDS	**VMC.KSDS.#1**	
	VMC.KSDS.#2	
ESDS	VMC.ESDS.#1	
AIX	VMC.AIX.#1	based on VMC.KSDS.#1
	VMC.AIX.#2	based on VMC.KSDS.#1
Path	VMC.PATH.#1	based on VMC.AIX.#1
	VMC.PATH.BCL	based on VMC.KSDS.#1

User Catalog

Name: VSAM.USER.CATALOG.A

Objects:

KSDS	**PAYROLL.FILE.BRANCH01**	
	PAYROLL.FILE.BRANCH02	
	•	
	•	
	PAYROLL.FILE.BRANCH25	
	PAYROLL.CONTROL.FILE1	
	PAYROLL.CONTROL.FILE2	
	VSAM.DATA.SET.CLUSTER	
	VSAM.DATA.SET.BCL	
	CALC.KSDS	
ESDS	CALC.ESDS	
	VSAM.DATA.FILE	
	PAYROLL.SUMMARY	
AIX	AIX.CALC.KSDS	based on CALC.KSDS
	VSAM.DATA.SET.AIX1	based on VSAM.DATA.SET.BCL
	VSAM.DATA.SET.AIX2	based on VSAM.DATA.SET.BCL
	VSAM.DATA.SET.AIX3	based on VSAM.DATA.SET.BCL
Path	VSAM.DATA.SET.P1	based on VSAM.DATA.SET.AIX1
	VSAM.DATA.SET.P2	based on VSAM.DATA.SET.AIX2
	PATH.CALC.AIX	based on AIX.CALC.KSDS

It is further assumed that the following DLBL statement has been placed into the permanent standard label area:

```
// DLBL IJSYSCT,'VSAM.MASTER.CATALOG',,VSAM
```

Backup Example 1

Objectives:

- Backup to tape.
- Backup of a single VSE/VSAM object from the master catalog.
- Automatic backup of the alternate indexes and paths based on the VSE/VSAM object.
- Automatic backup of the paths based on the alternate indexes of the VSE/VSAM object.
- Default buffer specification.

Specifications:

Catalog	**VSE/VSAM master catalog**
Entrynames	VMC.KSDS.#1
Buffers	Default number of buffers (3)
	Default buffer size
	Not fixed in real storage
Tape labels	None
Data compaction	No
Excluded objects	None

Job Stream:

```
// JOB         BACKUP1 BACKUP ONE OBJECT BY NAME
// ASSGN       SYS005,cuu
// EXEC        IDCAMS,SIZE=AUTO
   BACKUP      VMC.KSDS.#1
/*
/&
```

Objects Backed Up:

VMC.KSDS.#1	**specified VSE/VSAM object**
VMC.AIX.#1	alternate index for VMC.KSDS.#1
VMC.PATH.#1	path for VMC.AIX.#1
VMC.AIX.#2	alternate index for VMC.KSDS.#1
VMC.PATH.BCL	path based on VMC.KSDS.#1

Backup Example 2

Objectives:

- Backup to tape.
- Nongeneric backup of multiple VSE/VSAM objects from the job catalog.
- Specification of buffer parameters and options.
- Labeled backup file.

Specifications:

Catalog	**VSE/VSAM job catalog (VSAM.USER.CATALOG.A)**
Entrynames/password	CALC.KSDS
	CALC.ESDS

Catalog	VSE/VSAM job catalog (VSAM.USER.CATALOG.A)
	PAYROLL.CONTROL.FILE1/MPWD1
	PAYROLL.FILE.BRANCH01/MPWD2
Buffers	Number of buffers: 4
	Buffersize: 8192
	Fixed in real storage
Tape labels	BACKUP.FILE
Data compaction	No
Excluded objects	None

Job Stream:

```
// JOB    BACKUP2 BACKUP MULTIPLE OBJECTS BY NAME
// UPSI   1
// ASSGN  SYS005,cuu
// TLBL   TAPE,'BACKUP.FILE'
// DLBL   IJSYSUC,'VSAM.USER.CATALOG.A',,VSAM
// EXEC   IDCAMS,SIZE=AUTO
   BACKUP (CALC.KSDS CALC.ESDS         -
           PAYROLL.CONTROL.FILE1/MPWD1 -
           PAYROLL.FILE.BRANCH01/MPWD2 -
          )                            -
          BUFFERS(4)                   -
          BLOCKSIZE(8192)              -
          STDLABEL(TAPE)
/*
/&
```

Objects Backed Up:

CALC.KSDS	specified VSE/VSAM object
AIX.CALC.KSDS	alternate index for CALC.KSDS
PATH.CALC.AIX	path for AIX.CALC.KSDS
CALC.ESDS	specified VSE/VSAM object
PAYROLL.CONTROL.FILE1	specified VSE/VSAM object
PAYROLL.FILE.BRANCH01	specified VSE/VSAM object

Note: Instead of the two entry names CALC.KSDS and CALC.ESDS, you could simplify the specification by using the generic name CALC.*.

Backup Example 3

Objectives:

- Backup to disk.
- Generic backup of objects from the job catalog.
- No objects excluded.

Specifications:

Catalog	VSE/VSAM job catalog (VSAM.USER.CATALOG.A)
Entrynames	CALC.*
	VSAM.DATA.SET.*
Buffers	Default number of buffers (3)

Catalog	**VSE/VSAM job catalog (VSAM.USER.CATALOG.A)**
	Default buffer size
	Not fixed in real storage
Data compaction	No
Excluded objects	None

Job Stream:

```
// JOB    BACKUP3 BACKUP BY GENERIC NAME
// ASSGN  SYS005,cuu
// DLBL   BACKOUT,'BACKUP.FILE3',,SD
// EXTENT SYS005,DSK001,1,0,150,10
// DLBL   IJSYSUC,'VSAM.USER.CATALOG.A',,VSAM
// EXEC   IDCAMS,SIZE=AUTO
   BACKUP (CALC.* VSAM.DATA.SET.*)
/*
/&
```

Objects Backed Up:

CALC.ESDS	**specified via CALC.***
CALC.KSDS	specified via CALC.*
AIX.CALC.KSDS	alternate index for CALC.KSDS
PATH.CALC.AIX	path for AIX.CALC.KSDS
VSAM.DATA.SET.BCL	specified via VSAM.DATA.SET.*
VSAM.DATA.SET.AIX1	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.P1	path for VSAM.DATA.SET.AIX1
VSAM.DATA.SET.AIX2	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.P2	path for VSAM.DATA.SET.AIX2
VSAM.DATA.SET.AIX3	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.CLUSTER	specified via VSAM.DATA.SET.*

Backup Example 4

Objective:

- Backup to a disk of type FBA.
- Backup of all VSE/VSAM objects of the job catalog, with the exception of a generic group of objects and one other object.

Specifications:

Catalog	**VSE/VSAM job catalog (VSAM.USER.CATALOG.A)**
Entryname	*
Buffers	Default number of buffers
	Default buffer size
	Not fixed in real storage
Data compaction	Yes

Excluded objects	**PAYROLL.FILE.BRANCH01**
	•
	•
	PAYROLL.FILE.BRANCH25
	CALC.KSDS

Job Stream:

Permanent assignment of SYS005 is assumed for this example.

```
// JOB    BACKUP4 ALL OBJECTS OF CATALOG WITH GENERIC EXCLUSION
// ASSGN  SYS005,cuu
// DLBL   IJSYSUC,'VSAM.USER.CATALOG.A',,VSAM
// DLBL   BACKOUT,'BACKUP.FILE4',,SD
// EXTENT SYS005,DSKFB1,1,0,90,100
// EXEC   IDCAMS,SIZE=AUTO
   BACKUP (*/CATMPWD) -
          EXCL (PAYROLL.FILE.* CALC.KSDS) COMPACT
/*
/&
```

Objects Backed Up:

AIX.CALC.KSDS	**specified via ***
PATH.CALC.AIX	path for AIX.CALC.KSDS
CALC.ESDS	specified via *
PAYROLL.CONTROL.FILE1	specified via *
PAYROLL.CONTROL.FILE2	specified via *
PAYROLL.SUMMARY	specified via *
VSAM.DATA.FILE	specified via *
VSAM.DATA.SET.BCL	specified via *
VSAM.DATA.SET.AIX1	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.P1	path for VSAM.DATA.SET.AIX1
VSAM.DATA.SET.AIX2	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.P2	path for VSAM.DATA.SET.AIX2
VSAM.DATA.SET.AIX3	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.CLUSTER	specified via *

Note: AIX.CALC.KSDS and PATH.CALC.AIX are backed up despite the fact that CALC.KSDS is excluded from backup. This is because AIX.CALC.KSDS is "explicitly" specified by the generic name "*" causing its backup. As a result, PATH.CALC.AIX is also backed up.

If they are to be excluded from backup, AIX.CALC.KSDS must also be specified in the EXCLUDE parameter. PATH.CALC.AIX need not be specified in the EXCLUDE parameter because it is not a VSE/VSAM object and thus is only backed up if the alternate index on which it is based is backed up.

Backup Example 5

Objectives:

- Backup to disk; three backup extents on two volumes.
- Generic backup from the job catalog.
- Exclusion of one object from backup.

Specifications:

Catalog	**VSE/VSAM job catalog (VSAM.USER.CATALOG.A),**
	DLBL statement in permanent standard label area
Entryname	VSAM.DATA.*
Buffers	Default number of buffers
	Default buffer size
	Not fixed in real storage
Data Compaction	No
Excluded objects	VSAM.DATA.SET.CLUSTER

Job Stream:

```
// JOB     BACKUP5
// ASSGN   SYS005,cuu
// ASSGN   SYS006,cuu
// DLBL    BACKOUT,'BACKUP.FILE5',,SD
// EXTENT  SYS005,DSK001,1,0,165,10
// EXTENT  SYS005,DSK001,1,1,180,15
// EXTENT  SYS006,DSK002,1,2,165,20
// EXEC    IDCAMS,SIZE=AUTO
   BACKUP  (VSAM.DATA.*) -
           EXCLUDE (VSAM.DATA.SET.CLUSTER)
/*
/&
```

Objects Backed Up:

VSAM.DATA.FILE	**specified via VSAM.DATA.***
VSAM.DATA.SET.BCL	specified via VSAM.DATA.*
VSAM.DATA.SET.AIX1	alternate index of VSAM.DATA.SET.BCL
VSAM.DATA.SET.P1	path for VSAM.DATA.SET.AIX1
VSAM.DATA.SET.AIX2	alternate index of VSAM.DATA.SET.BCL
VSAM.DATA.SET.P2	path for VSAM.DATA.SET.AIX2
VSAM.DATA.SET.AIX3	alternate index of VSAM.DATA.SET.BCL

Backup Example 6

Objective:

- Backup to tape.
- Backup all VSE/VSAM objects of the master catalog.

Specifications:

Catalog	**VSE/VSAM master catalog**
Entryname	*
Buffers	Default number of buffers
	Default buffer size
	Not fixed in real storage
Tape labels	None
Data compaction	Yes

Catalog	VSE/VSAM master catalog
Excluded objects	None

Job Stream: Permanent assignment of SYS005 is assumed for this example.

```
// JOB     BACKUP6
// EXEC    IDCAMS,SIZE=AUTO
   BACKUP  (*) COMPACT
/*
/&
```

Objects Backed Up:

VMC.ESDS.#1	specified via *
VMC.KSDS.#1	specified via *
VMC.AIX.#1	alternate index of VMC.KSDS.#1
VMC.PATH.#1	path for VMC.AIX.#1
VMC.AIX.#2	alternate index of VMC.KSDS.#1
VMC.PATH.BCL	path for VMC.KSDS.#1
VMC.KSDS.#2	specified via *

Backup Example 7

Objective:

- Backup to tape.
- Backup all VSE/VSAM objects of the master catalog and a user catalog to the same tape.

Specifications:

Catalog	VSE/VSAM master catalog and user catalog
Entryname	* and VSAM.USER.CATALOG.A
Buffers	Default number of buffers
	Default buffer size
	Not fixed in real storage
Tape labels	Standard
Data compaction	Yes
Excluded objects	None

Job Stream: Permanent assignment of SYS005 is assumed for this example.

```
// JOB     BACKUP7
// TLBL    TAPE1,'BACKUP.FILE1'
// EXEC    IDCAMS,SIZE=AUTO
   BACKUP  (*) COMPACT                  -
           STDLABEL(TAPE1)              -
           NOREWIND
/*
// TLBL    TAPE2,'BACKUP.FILE2'
// EXEC    IDCAMS,SIZE=AUTO
   BACKUP  (VSAM.USER.CATALOG.A) COMPACT -
           STDLABEL(TAPE2)              -
           NOREWIND
/*
/&
```

Objects Backed Up:

VMC.ESDS.#1	specified via *
VMC.KSDS.#1	specified via *
VMC.AIX.#1	alternate index of VMC.KSDS.#1
VMC.PATH.#1	path for VMC.AIX.#1
VMC.AIX.#2	alternate index of VMC.KSDS.#1
VMC.PATH.BCL	path for VMC.KSDS.#1
VMC.KSDS.#2	specified via *
VSAM.USER.CATALOG.A	specified explicitly

Backup Example 8

Objective:

- Back up snapped volumes after FlashCopy on ESS.

The following jobstream example shows how to:

1. Create a snapshot of the *source* volumes *SYSWK1* and *DOSRES* to the *snapped (target)* volumes *VOLSN1* and *VOLSN2*.
2. Run an IDCAMS IMPORT CONNECT to inform the z/VSE system that a copy of the user catalog (VSESP.USER.CATALOG on SYSWK1 and DOSRES) now exists on the snapped (target) volume, and that this copy of the user catalog now has a *synonym name* (VSESP.SNAP.CATALOG).
3. Run an IDCAMS BACKUP that uses the parameters contained in the synonym list.

Specifications:

Catalog	VSE/VSAM master catalog and user catalog
Entryname	Volume list
Buffers	Default number of buffers
	Default buffer size
	Not fixed in real storage
Tape labels	Standard
Data compaction	Yes
Excluded objects	None

Job Stream:.

```
// JOB SNAP and BACKUP from snapped Volumes
// ASSGN SYS005,180
// DLBL IJSYSUC,'VSESP.SNAP.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 /* First: do the SNAPSHOT */
  SNAP                                          -
     SOURCEVOLUMES(SYSWK1 DOSRES)               -
     TARGETVOLUMES(VOLSN1 VOLSN2)
 /* Second: Synonym Name for the snapped Catalog */
  IMPORT CONNECT OBJECTS((VSESP.SNAP.CATALOG     -
     VOLUMES(VOLSN1) DEVT(3390)))               -
     CATALOG(VSAM.MASTER.CATALOG)
 /* Third: Backup from snapped volumes */
  BACKUP (*)                                    -
     SYNONYMLIST(                               -
     SOURCEVOLUMES(SYSWK1 DOSRES)               -
     TARGETVOLUMES(VOLSN1 VOLSN2)                  -
     CATALOG(VSESP.USER.CATALOG)                -
     SYNCATALOG(VSESP.SNAP.CATALOG)     )
```

```
/*
/&
```

Objects Backed Up:

VSESP.USER.CATALOG	**specified via * and synonym list**

Examples of Functions for RESTORE

Generic Names

As with the BACKUP command, you can specify either individual objects or a generic name, with or without the EXCLUDE parameter (see “Generic Names” on page 242). Note that the meaning of a generic name is slightly different for RESTORE. For RESTORE, the generic name applies to the objects of the backup file, rather than to the objects of a catalog as is the case for BACKUP. Figure 19 on page 253 shows an example of restoring objects with a generic name.

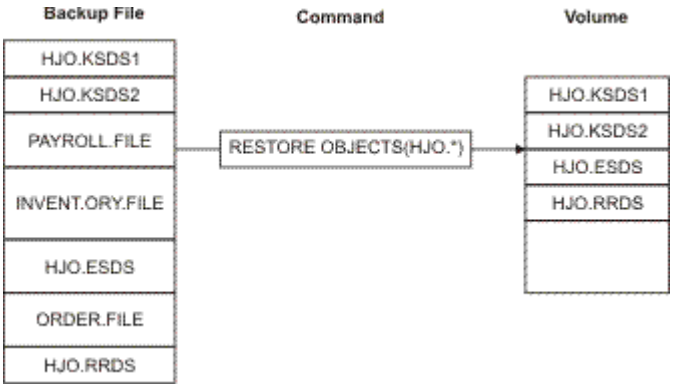


Figure 19. Generic Restoration

Excluding Objects from Restoration

Excluding objects from the entryname list by means of the EXCLUDE parameter causes automatic exclusion of any associations of those objects. If, however, an association is a member of the candidate list (specified in the OBJECTS parameter), it is restored even if its base cluster is excluded from restoration. See Figure 20 on page 253.

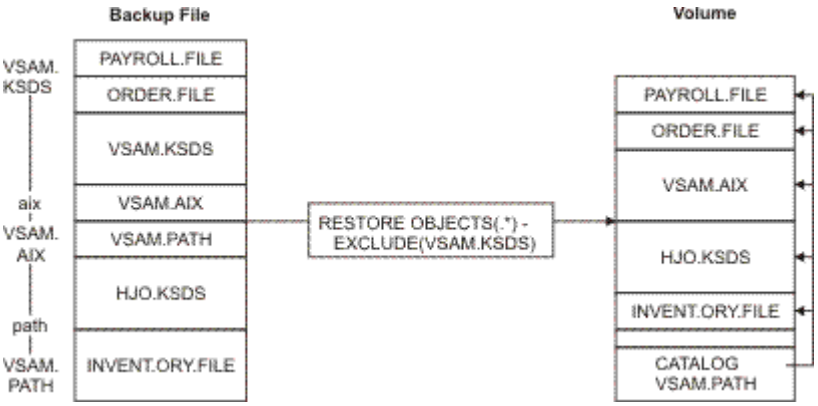


Figure 20. Restoration of an Alternate Index Excluding the Base Cluster

Restoration to a Volume of the Same Device Type

If a VSE/VSAM object is to be restored to the same volume(s) on which it was before, the location of the restored suballocatable files on the volume(s) is determined by VSE/VSAM. You can, however, influence

the location of such files through the USECLASS, DATAUSECLASS, and INDEXUSECLASS parameters. These parameters allow you to select space classes (for the data and/or index component of an object) during restoration that are different from the originally chosen space classes.

An important application of these parameters is the following: Assume that the original space class represented a fixed-head area that still contains some VSE/VSAM files, but that its remaining free space (possibly the space from which the file was backed up) is damaged. Without moving the intact files contained in the fixed-head space, the data space as such cannot be deleted. Nor does VSE/VSAM mark the damaged space as defective. If you do not change the space class, VSE/VSAM tries to suballocate the space into which the file is to be restored out of the "damaged space," and the restoration will be unsuccessful. With the USECLASS, DATAUSECLASS, and INDEXUSECLASS subparameters, the file or component in question can be taken out of the fixed-head space class and moved to another, nondefective space class. For an example, see “Restore Example 9” on page 266.

For *unique* files, you must specify, in the RESTORE command, the filename of the DLBL/EXTENT statements indicating the location where you want your files to be restored. The location may be the same as the original location or it may be a different one. For an example, see “Restore Example 3” on page 261.

Restoration with File Modifications

You can make any or all of the following modifications to your files when you restore them:

- Move files to a space of a different use class;
- Move files to a volume of a different device type;
- Change the allocation size of the data component of a specific file;
- Change the size of the index CI of a specific file.

Specifying a new use class has no appreciable effect on the performance of the RESTORE command. If you request any of the other file modifications, however, one or more of the following attributes of the cluster is likely to change:

- CA size
- Block size (automatic)
- Index CI size
- Space allocation size.

Because of these possible modifications, you should not indiscriminately migrate files from one device to another, or change the data component allocation size or index CI size. These file modifications can result in degraded performance during RESTORE execution and changed space allocation sizes due to the new device characteristics.

Restoration to a Volume of a Different Device Type

To restore a file to a volume of different device type, simply specify *volser* in the VOLUMES subparameter (or the DATAVOLUMES or INDEXVOLUMES subparameters) to represent the new volume. You can copy your files onto volumes of a different device type without specifying VOLUMES if one of the following situations has occurred:

- *Volser* has been assigned to a volume of a different device type since the BACKUP was performed; or
- The controlling catalog (during restoration) owns a volume of a different device type that contains the same *volser.*

The only restriction in moving files to a different device is that the data CI size must not be changed in the transition. This is necessary to avoid record-level reorganization, with its undesirable performance characteristics. Spanned ESDSs cannot be restored to a volume of a different device type if a change in the data component CA size would be required.

Be aware that if you

- Back up a file on device type A,
- Restore it to device type B,
- Back it up from B,
- Restore it to A,

it will probably no longer fit into its original storage allocation on device A. This is because VSE/VSAM generally needs to round up the storage allocation request when moving objects from one device type to another.

Using RESTORE to Add Entries to a Catalog

Backup/Restore processes both the catalog information for an object and the object itself. This means that you can add objects to a catalog simply by specifying the name of the new catalog at restoration. VSE/VSAM will copy the catalog information for the objects and the objects themselves. For empty objects, only the catalog information will be copied. If the new catalog already contains an entry with the same name as the object being restored, that entry is deleted. Based on the information in the tape backup file, a new object is defined with the same name, and the file is restored.

Be aware that an object may be defined in more than one catalog at once. If you intend to move (not just copy) an object from one catalog to another catalog, you must delete the object from the source catalog after using BACKUP and RESTORE to copy the object's information into the new catalog.

You can merge objects defined in several different catalogs into one catalog by restoring the objects in the order you want them to be defined in the new catalog.

Subsetting During Restoration

The RESTORE command allows subparameters (such as the VOLUMES, DATAVOLUMES, or the USECLASS subparameters) for the individual entrynames. For a generic name, the specified subparameters apply to all objects of the generic name.

It is possible to exempt objects represented by a generic name from the subparameters. This can be achieved by specifying these objects with their respective subparameters in addition to the generic name. The subparameters of the generic name then apply to all objects represented by the generic name except those explicitly named with other subparameters.

For example, if the volume serial number ABCVOL is specified for the entryname ABC.* and VOLXYZ is specified for ABC.KSDS.*, then the object ABC.KSDS.#1 is restored onto the volume VOLXYZ whereas the object ABC.ESDS is restored onto ABCVOL. For an example, see “Restore Example 2” on page 259.

This technique is called subsetting and applies to all subparameters, including passwords for an entryname. If multiple entrynames form subsets of one another, the subparameters applicable to an object are those specified for the entryname that is closest ("best fit") to the object's name.

Subparameters (except passwords) never apply to objects that are automatically restored. If you want to change the volumes or the useclass of such objects, you must name them in the OBJECTS parameter together with the subparameters you wish to apply.

Reorganization of CAs

The restoration of a KSDS causes a reorganization of the data component on the basis of CAs. The individual records of a CI and the CIs of a CA are not reorganized, but the CAs as a whole are. Control areas that previously were not physically adjacent because of CA splits will be physically adjacent after restoration, thus avoiding unnecessary disk arm movements for sequential processing.

The reorganization of KSDS CAs as a whole results in a change of the relative byte addresses.

Considerations: Restoration from a Tape Resident Backup File

For restoration, the backup file must always be assigned to SYS004 as for IMPORT.

The interspersed tape marks of the backup file allow individual objects that are not to be restored to be skipped during restoration. In this way the tape channel can be freed for the duration of the skipping operation.

When all specified files have been restored from a backup volume, the tape is unloaded.

Selective Tape Mounting for Restoration

The restoration function does not require that the first volume of the backup file be mounted when the execution of the RESTORE command begins.

Every volume of the backup file contains a directory by means of which Backup/Restore determines which of the VSE/VSAM objects to be restored are on the mounted (or later) volumes of the backup file and which are on earlier volumes.

VSE/VSAM Backup/Restore restores the specified VSE/VSAM objects in the order in which they are stored on the backup file. It starts at the beginning of the mounted volume with the first object that is not an association, proceeds to the end of the backup file, and then continues with the first volume of the backup file (wrap-around). The initially mounted volume has to be mounted again if it contains part of an object that starts on an earlier volume, or if it starts with an associated object.

When the operator is asked to mount a later volume, he may skip any volumes of the backup file that do not contain objects to be restored. The cross-reference listing printed upon completion of the BACKUP command will help him to determine the volumes to be mounted.

For VSE/VSAM objects that start on earlier volumes than the initially mounted one, VSE/VSAM Backup/Restore knows the exact volume and asks the operator for this volume. For the correct tape volume to be mounted, a message is printed specifying the volume sequence number assigned by the BACKUP command and listed in the cross-reference listing. Restoration continues after the operator has mounted the correct tape volume.

You do not lose the ability to skip unwanted objects by not mounting the first tape volume of the backup file.

All volumes of the backup file contain the backup file creation time. Whenever a new volume is mounted, it is ensured that the new and the old volumes have the same backup file creation time stamp. Otherwise, an error message is issued that asks the operator for the correct volume. Processing can proceed after the correct volume has been mounted.

Sequential Tape Mounting for Restoration

A VSE/VSAM object to be restored may span several tape volumes of the backup file. When processing of a volume has been completed, the volume is rewound and unloaded, and processing continues with the next part of the object.

Every tape volume of the backup file terminates with an end-of-tape (EOT) record containing a volume termination time stamp that is identical with the volume creation time stamp of the next tape volume. The backup *volume* creation time stamp is in addition to the backup *file* creation time stamp. When an object spans multiple tape volumes and a transition from one tape volume to the next takes place, the termination time stamp of the volume just completed and the volume creation time stamp of the next volume are compared. If they do not match, an error message is issued that requests mounting of the correct volume. Processing can proceed after the correct volume has been mounted.

Alternate Tape Support for Restoration

If you assign alternate tapes for restoration, processing will automatically switch to the alternate tape drive and back as the individual volumes are processed. See Figure 21 on page 258.

However, if VSE/VSAM objects are restored selectively, the program may ask the operator to mount a later volume (message IDC402A). It is recommended to mount the last backup volume containing all information about the VSE/VSAM objects not yet restored. The program now informs the operator which volume must be mounted next (message IDC400A).

Considerations: Restoration from a Disk Resident Backup File

For restoration from disk, the backup file must be assigned to SYS004. If the backup file extends over several disks, the corresponding devices must be assigned to consecutive logical units, beginning with SYS004.

Assignments are needed only for those volumes that contain objects that you want to restore.

The DLBL must contain the standard file name BACKIN, and you have to provide the file-ID of the backup file. The file type code in the DLBL statement must be DA (because the backup file is processed by the RESTORE program as a file of the type DA). For an example, refer to "Restore Example 2" on page 259.

For each backup volume, you need to specify only one EXTENT statement. You can also selectively restore backed up objects in any sequence. These advantages are possible because:

- All extents belonging to the backup file on a backup file volume are found by means of the file labels contained in the corresponding VTOC.
- The first extent on each volume contains the backup file directory.

  This directory contains comprehensive information on the location of each backed up object.

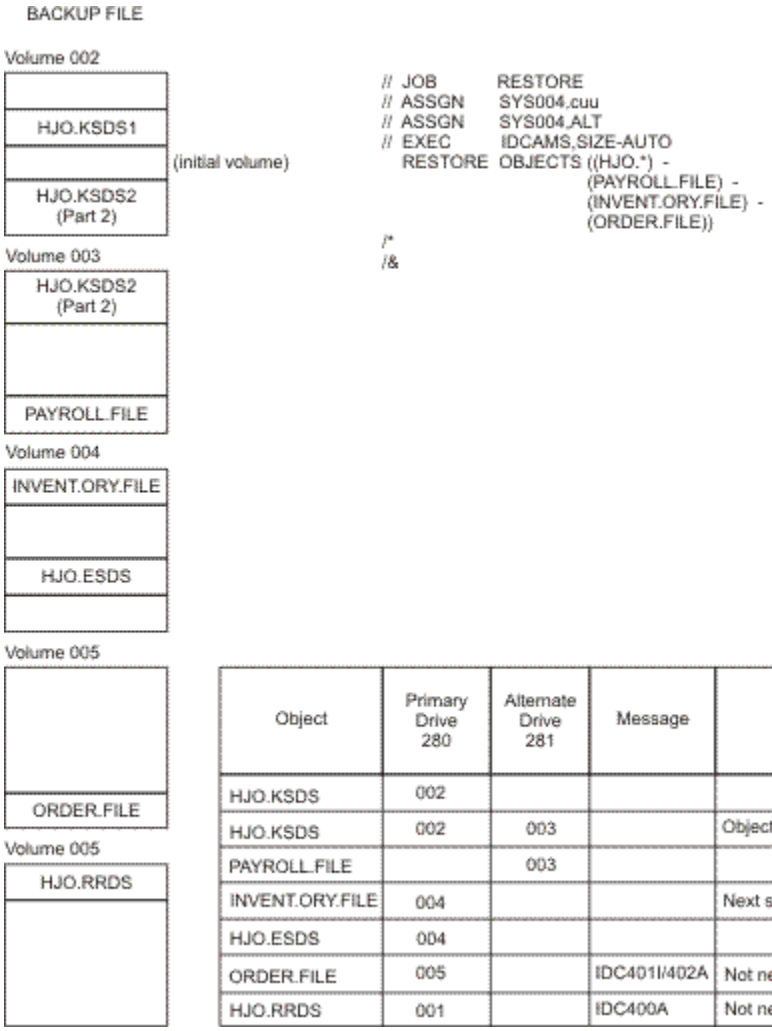
The end of an object in a backup file on CKD and ECKD disks is marked with an EOF record. Therefore, the backup file cannot be processed as a normal file; it must be processed by means of VSE/VSAM Backup/Restore. The restoration of VSE/VSAM objects from a disk resident backup file is quite simple and does not require operator intervention.

Buffer Allocation for Restoration

With restoration, VSE/VSAM Backup/Restore automatically chooses a buffer size equal to the appropriate buffer size for backup. No user specification is required.

The BUFFERS parameter can, however, be used to influence the number of buffers available for restoration. This parameter is intended to assist streaming mode of a tape device in a multiprogramming environment. The maximum number of buffers used for restoration from a tape resident backup file is that used for backup. If a larger number is specified, it is reduced to the permitted maximum.

For a backup file *on disks*, any number of buffers greater than 3 can be specified. However, specifying a high number of buffers does not increase the performance of the restoration process, because read operations for a disk resident backup file can only be performed one after the other (that is, read operations cannot overlap one another).



Object	Primary Drive 280	Alternate Drive 281	Message	Remarks
HJO.KSDS	002			
HJO.KSDS	002	003		Object spans volumes
PAYROLL.FILE		003		
INVENT.ORY.FILE	004			Next sequential object
HJO.ESDS	004			
ORDER.FILE	005		IDC401I/402A	Not next object
HJO.RRDS	001		IDC400A	Not next object,001 earlier volume

Figure 21. Alternate Tape Support for Restoration

Restore Job Examples

The following restoration examples are based on the backup files created by the examples shown under “Backup Job Examples” on page 244. Reference to the backup file is made by the backup job name (BACKUP3) used in the backup examples. The meaning of the objects is the same as defined under “Backup Job Examples” on page 244.

It is assumed that the DLBL statement for the VSE/VSAM master catalog is contained in the permanent standard label area.

Restore Example 1

Objectives:

- Nongeneric restoration of multiple VSE/VSAM objects into the job catalog.
- Specification of the buffer parameters and options.
- Restoration from tape.
- No define modifications.
- All VSE/VSAM objects suballocatable.

Specifications:

Catalog	VSE/VSAM job catalog (VSAM.USER.CATALOG.A)
Backup file	Created by BACKUP2
Entrynames/password	AIX.CALC.KSDS
	PAYROLL.CONTROL.FILE1/MPWD1
Buffers	Number of buffers: 2
Not fixed in real storage	
Tape labels	BACKUP.FILE
Excluded objects	None
Modifications	None

Job Stream:

```
// JOB     RESTORE1
// UPSI    1
// ASSGN   SYS004,cuu
// TLBL    TAPE,'BACKUP.FILE'
// DLBL    IJSYSUC,'VSAM.USER.CATALOG.A',,VSAM
// EXEC    IDCAMS,SIZE=AUTO
   RESTORE OBJECTS((AIX.CALC.KSDS)                -
                   (PAYROLL.CONTROL.FILE1/MPWD1)) -
           BUFFERS(2)                             -
           STDLABEL(TAPE)
/*
/&
```

Restored Objects:

AIX.CALC.KSDS	specified VSE/VSAM object
PATH.CALC.AIX	path for AIX.CALC.KSDS
PAYROLL.CONTROL.FILE1	specified VSE/VSAM object

Restore Example 2

Objectives:

- Restoration of all VSE/VSAM objects of a backup file.
- Restoration from an FBA disk.
- Restoration of a generic group of objects to a new set of volumes, with the exception of two objects of the generic group that are restored to different volumes.
- All VSE/VSAM objects suballocatable.

Specifications:

Catalog	VSE/VSAM job catalog, no passwords
	DLBL statement in permanent standard label area
Backup file	Created by BACKUP4
Entrynames	*with no objects excluded
	VSAM.DATA.*
	VSAM.DATA.SET.BCL
	VSAM.DATA.SET.CLUSTER
Buffers	Default number of buffers
	Not fixed in real storage

Catalog	VSE/VSAM job catalog, no passwords
Modifications	Data components of VSAM.DATA.SET.BCL and VSAM.DATA.SET.CLUSTER to be restored onto volume DATVOL Index components of VSAM.DATA.SET.BCL and VSAM.DATA.SET.CLUSTER to be restored onto volume IXVOL Data component of all other objects of VSAM.DATA.* to be restored onto VOL001 Index component of all other objects of VSAM.DATA.* to be restored onto VOL002
Job control	Permanent assignment for SYS004

Job Stream:

```
// JOB       RESTORE2
// ASSGN     SYS004,cuu
// DLBL      BACKIN,'BACKUP.FILE4',,DA
// EXTENT    SYS004,DSKFB1
// EXEC      IDCAMS,SIZE=AUTO
   RESTORE OBJECTS((*)                        -
                   (VSAM.DATA.*               -
                        DATAVOLUMES(VOL001)   -
                        INDEXVOLUMES(VOL002)  -
                   )                          -
                   (VSAM.DATA.SET.BCL         -
                        DATAVOLUMES(DATVOL)   -
                        INDEXVOLUMES(IXVOL)   -
                   )                          -
                   (VSAM.DATA.SET.CLUSTER     -
                        DATAVOLUMES(DATVOL)   -
                        INDEXVOLUMES(IXVOL)
                   )                          -
                  )
/*
/&
```

Restored Objects:

AIX.CALC.KSDS	onto original volumes
PATH.CALC.AIX	
CALC.ESDS	onto original volumes
PAYROLL.CONTROL.FILE1	onto original volumes
PAYROLL.CONTROL.FILE2	onto original volumes
PAYROLL.SUMMARY	onto original volumes
VSAM.DATA.FILE	data component onto VOL001
VSAM.DATA.SET.BCL	data component onto DATVOL index component onto IXVOL
VSAM.DATA.SET.AIX1	data component onto VOL001 index component onto VOL002
VSAM.DATA.SET.P1	
VSAM.DATA.SET.AIX2	data component onto VOL001

AIX.CALC.KSDS	**onto original volumes**
	index component onto VOL002
VSAM.DATA.SET.P2	
VSAM.DATA.SET.AIX3	data component onto VOL001
	index component onto VOL002
VSAM.DATA.SET.CLUSTER	data component onto DATVOL
	index component onto IXVOL

Restore Example 3

Objective:

- Restoration of VSE/VSAM objects with a UNIQUE data component.
- Restoration from disk.

Specifications:

Catalog	**VSE/VSAM master catalog, no passwords**
Backup file	Created by BACKUP3
Entrynames	CALC.KSDS
Buffers	Default buffer number
	Default buffer size
	Not fixed in real storage
Excluded objects	None
Modifications	None

Job Stream:

```
// JOB     RESTORE3
// ASSGN   SYS004,cuu
// DLBL    BACKIN,'BACKUP.FILE3',,DA
// EXTENT  SYS004,DSK001
// DLBL    DATA,,,VSAM
// EXTENT  ,VOL111,,,95,90
// EXEC    IDCAMS,SIZE=AUTO
   RESTORE OBJECTS(CALC.KSDS -
                   DATAFILE(DATA))
/*
/&
```

Restored Objects:

CALC.KSDS	**specified object**
AIX.CALC.KSDS	alternate index for CALC.KSDS
PATH.CALC.AIX	path for AIX.CALC.KSDS

Restore Example 4

Objectives:

- Restoration of a single VSE/VSAM object of the backup file into the master catalog.
- Automatic restoration of the alternate indexes and paths based on that object.
- Automatic restoration of the paths based on the alternate indexes of the VSE/VSAM object.

- Default buffer specifications.
- All VSE/VSAM objects suballocatable.
- Restoration from tape.

Specifications:

Catalog	VSE/VSAM master catalog
Backup file	Created by BACKUP6
Entrynames	VMC.KSDS.#1
Buffers	Default number of buffers
	Not fixed in real storage
Tape labels	None
Excluded objects	None
Modifications	None

Job Stream:

```
// JOB      RESTORE4
// ASSGN    SYS004,cuu
// EXEC     IDCAMS,SIZE=AUTO
   RESTORE  OBJECTS(VMC.KSDS.#1)
/*
/&
```

Restored Objects:

VMC.KSDS.#1	specified VSE/VSAM object
VMC.AIX.#1	alternate index of VMC.KSDS.#1
VMC.PATH.#1	path for VMC.AIX.#1
VMC.AIX.#2	alternate index of VMC.KSDS.#1
VMC.PATH.BCL	path for VMC.KSDS.#1

Restore Example 5

Objectives:

- Generic restoration of multiple VSE/VSAM objects into a specified VSE/VSAM catalog.
- No objects excluded.
- All VSE/VSAM objects suballocatable.
- Restoration from disk.

Specifications:

Catalog	Explicitly specified catalog (VSAM.CATALOG)
Backup file	Created by BACKUP3
Entrynames	VSAM.*
	CALC.*
Buffers	Default number of buffers
	Not fixed in real storage
Excluded objects	None

Catalog	Explicitly specified catalog (VSAM.CATALOG)
Modifications	None

Job Stream:

```
// JOB      RESTORE5
// ASSGN    SYS004,cuu
// DLBL     BACKIN.'BACKUP.FILE3',,DA
// EXTENT   SYS004,DSK001
// EXEC     IDCAMS,SIZE=AUTO
   RESTORE  OBJECTS((VSAM.*)(CALC.*)) -
            CATALOG(VSAM.CATALOG)
/*
/&
```

Restored Objects:

CALC.ESDS	specified via CALC.*
CALC.KSDS	specified via CALC.*
AIX.CALC.KSDS	alternate index of CALC.KSDS
PATH.CALC.AIX	path for AIX.CALC.KSDS
VSAM.DATA.SET.BCL	specified via VSAM.*
VSAM.DATA.SET.AIX1	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.P1	path for VSAM.DATA.SET.AIX1
VSAM.DATA.SET.AIX2	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.P2	path for VSAM.DATA.SET.AIX2
VSAM.DATA.SET.AIX3	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.CLUSTER	specified via VSAM.*

Note: Because the two generic names VSAM.* and CALC.* make up all objects of the backup file, you could simplify the specification by using the generic name "*".

Restore Example 6

Objectives:

- Generic restoration into the job catalog.
- Exclusion of one VSE/VSAM object from restoration.
- No define modifications.
- All VSE/VSAM objects suballocatable.
- Restoration from backup file extents, the extents being on two disk volumes.

Specifications:

Catalog	Job catalog (VSAM.USER.CATALOG.A), DLBL statement in permanent standard label area, No passwords
Backup file	Created by BACKUP5
Entrynames	VSAM.DATA.SET.*
Buffers	Default number of buffers

Catalog	**Job catalog (VSAM.USER.CATALOG.A),**
	Not fixed in real storage
Excluded objects	**VSAM.DATA.SET.AIX1**
Modifications	None

Job Stream:

```
// JOB      RESTORE6
// ASSGN    SYS004,cuu
// ASSGN    SYS005,cuu
// DLBL     BACKIN,'BACKUP.FILE5',,DA
// EXTENT   SYS004,DSK001
// EXTENT   SYS004,DSK001
// EXTENT   SYS005,DSK002
// EXEC     IDCAMS,SIZE=AUTO
   RESTORE  OBJECTS(VSAM.DATA.SET.*) -
            EXCLUDE(VSAM.DATA.SET.AIX1)
/*
/&
```

Restored Objects:

VSAM.DATA.SET.BCL	**specified via VSAM.DATA.SET.***
VSAM.DATA.SET.AIX2	alternate index of VSAM.DATA.SET.BCL
VSAM.DATA.SET.P2	path for VSAM.DATA.SET.AIX2
VSAM.DATA.SET.AIX3	alternate index of VSAM.DATA.SET.BCL

Note: VSAM.DATA.SET.P1 is not restored because it is based upon VSAM.DATA.SET.AIX1 and is not a VSE/VSAM object. By the redefinition of VSAM.DATA.SET.BCL, existing versions of VSAM.DATA.SET.AIX1 and VSAM.DATA.SET.P1 are deleted.

Restore Example 7

Objectives:

- Restoration of all VSE/VSAM objects of a backup file into the master catalog.
- No define modifications.
- All VSE/VSAM objects suballocatable.
- Restoration from tape.

Specifications:

Catalog	**VSE/VSAM master catalog, no passwords**
Backup file	Created by BACKUP6
Entrynames	*
Buffers	Default number of buffers
	Not fixed in real storage
Tape labels	None
Excluded objects	None
Modifications	None
Job Control	Permanent assignment for SYS004

Job Stream:

```
//  JOB      RESTORE7
//  ASSGN    SYS004,cuu
//  EXEC     IDCAMS,SIZE=AUTO
    RESTORE  OBJECTS(*)
/*
/&
```

Restored Objects: All objects of backup file.

Restore Example 8

Objectives:

- Restoration of the VSE/VSAM objects of a backup file with the exception of a generic group of VSE/VSAM objects and one individual VSE/VSAM object.
- Restoration into the password-protected master catalog.
- No define modifications.
- All VSE/VSAM objects suballocatable.
- Restoration from an FBA disk.

Specifications:

Catalog	**VSE/VSAM master catalog (VSAM.MASTER.CATALOG),**
	update password = UPDPW
Backup file	Created by BACKUP4
Entrynames	*
Buffers	Default number of buffers
	Not fixed in real storage
Excluded objects	PAYROLL.CONTROL.FILE1
	PAYROLL.CONTROL.FILE2
	PAYROLL.SUMMARY
	AIX.CALC.KSDS
Modifications	None
Job Control	Permanent assignment for SYS004

Job Stream:

```
// JOB      RESTORE8
//  ASSGN    SYS004,cuu
// DLBL     BACKIN,'BACKUP.FILE4',,DA
// EXTENT   SYS004,DSKFB1
// EXEC     IDCAMS,SIZE=AUTO
   RESTORE  OBJECTS(*)                        -
            EXCLUDE(PAYROLL.* AIX.CALC.KSDS) -
            CATALOG(VSAM.MASTER.CATALOG/UPDPW)
/*
/&
```

Restored Objects:

CALC.ESDS	**specified via ***
VSAM.DATA.FILE	specified via *
VSAM.DATA.SET.BCL	specified via *

CALC.ESDS	specified via *
VSAM.DATA.SET.AIX1	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.P1	path for VSAM.DATA.SET.AIX1
VSAM.DATA.SET.AIX2	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.P2	path for VSAM.DATA.SET.AIX2
VSAM.DATA.SET.AIX3	alternate index for VSAM.DATA.SET.BCL
VSAM.DATA.SET.CLUSTER	specified via *

Restore Example 9

Objectives:

- Restoration of all VSE/VSAM objects of a backup file.
- Restoration of all objects to a common set of volumes that is different from the originating set of volumes.
- Restoration of all objects to a new space class.
- All VSE/VSAM objects suballocatable.
- Restoration from tape.

Specifications:

Catalog	VSE/VSAM job catalog, no passwords,
	DLBL statement in permanent standard
	label area
Backup file	Created by BACKUP6
Entrynames	*
Buffers	Default number of buffers
	Not fixed in real storage
Tape labels	None
Excluded objects	None
Modifications	Global change to volumes VOL001 and VOL002
	Global change to space class 7, primary
	and secondary
Job control	Permanent assignment for SYS004

Job Stream:

```
//  JOB      RESTORE9
//  ASSGN    SYS004,cuu
//  EXEC     IDCAMS,SIZE=AUTO
    RESTORE  OBJECTS(*)              -
             VOLUMES(VOL001 VOL002) -
             USECLASS(7 P)
/*
/&
```

Restored Objects: All objects of backup file.

Restore Example 10

Objectives:

- Restoration of objects from a tape that was backed up using VSE/VSAM Backup/Restore Release 1.
- Restoration to a device of a different device type than the objects were backed up from.

Specifications:

Catalog	VSE/VSAM master catalog
Backup file	Created by BACKUP6
Entrynames	*
Buffers	Default number of buffers
	Not fixed in real storage
Tape labels	None
Excluded objects	None
Modifications	Move objects to a volume of a different device type
Job control	Permanent assignment for SYS004

Job Stream:

```
//  JOB      RESTOR10
//  ASSGN    SYS004,cuu
//  EXEC     IDCAMS,SIZE=AUTO
    RESTORE  OBJECTS((*)     -
             VOLUMES(338001))
/*
/&
```

Restored Objects: All objects of backup file.

Restore Example 11

Objectives:

- Generic restoration of VSE/VSAM objects from tape backup file.
- Restoration to a volume of a different device type than the objects were backed up from.

Specifications:

Volume backed up from	**338001**
Volume restored to	337001
Catalog	VSE/VSAM master catalog
Backup file	Created by BACKUP6
Entrynames	VMC.KSDS.*
Buffers	Default number of buffers
	Not fixed in real storage
Tape labels	None
Excluded objects	None
Modifications	Generic restoration to a volume of a different device type

Job Stream:

```
//  JOB      RESTOR11
//  ASSGN    SYS004,cuu
//  EXEC     IDCAMS,SIZE=AUTO
    RESTORE  OBJECTS((VMC.KSDS.* -
             VOLUMES(337001)))
/*
/&
```

Restored Objects:

VMC.KSDS.#1	**specified via VMC.KSDS.*, restored to 337001**
VMC.AIX.#1	alternate index of VMC.KSDS.#1, restored to 338001
VMC.AIX.#2	alternate index of VMC.KSDS.#1, restored to 338001
VMC.PATH.#1	path for VMC.AIX.#1, restored to 338001
VMC.PATH.BCL	path for VMC.KSDS.#1, restored to 338001
VMC.KSDS.#2	specified via VMC.KSDS.*, restored to 337001

Note: To restore the AIXes and paths to the new device type, specify VMC.AIX.* and VMC.PATH.* as *entrynames*.

Restore Example 12

Objectives:

- Restoration of a specific KSDS from the job catalog.
- Increasing the storage allocation for the KSDSs data component.
- Restoration from disk.

Specifications:

Catalog	**Job catalog (VSAM.USER.CATALOG.A), no passwords**
Backup file	Created by BACKUP3
Entrynames	VSAM.DATA.SET.BCL
Buffers	Default number of buffers
	Not fixed in real storage
Excluded objects	None
Modifications	Increase the storage allocation for the data component

Job Stream:

```
//  JOB      RESTOR12
//  ASSGN    SYS004,cuu
// DLBL      BACKIN,'BACKUP.FILE3',,DA
// EXTENT    SYS004,DSK001
//  DLBL     IJSYSUC,'VSAM.USER.CATALOG.A',,VSAM
//  EXEC     IDCAMS,SIZE=AUTO
    RESTORE  OBJECTS((VSAM.DATA.SET.BCL -
             DATARECORDS(400 50)))
/*
/&
```

Restored Objects:

VSAM.DATA.SET.BCL	**The data component of VSAM.DATA.SET.BCL is restored to an area with a primary allocation of 400 records and a secondary allocation of 50 records.**
VSAM.DATA.SET.AIX1	AIX based on VSAM.DATA.SET.BCL, restored to whatever its original storage allocation was.
VSAM.DATA.SET.AIX2	AIX based on VSAM.DATA.SET.BCL, restored to whatever its original storage allocation was.
VSAM.DATA.SET.AIX3	AIX based on VSAM.DATA.SET.BCL, restored to whatever its original storage allocation was.
VSAM.DATA.SET.P1	Path based on VSAM.DATA.SET.AIX.1, restored to whatever its original storage allocation was.
VSAM.DATA.SET.P2	Path based on VSAM.DATA.SET.AIX.2, restored to whatever its original storage allocation was.

Restore Example 13

Objectives:

- Generic restoration of VSE/VSAM objects from a two-file tape backup.
- Restoration to a volume of a different device type than the objects were backed up from.

Specifications:

Volume backed up from	**338001**
Volume restored to	337001
Catalog	VSE/VSAM master catalog and user catalog
Backup file	Created by BACKUP7
Entrynames	VMC.KSDS.*
Buffers	Default number of buffers Not fixed in real storage
Tape labels	Standard
Excluded objects	None
Modifications	Generic restoration to a volume of a different device type

Job Stream:

```
//  JOB      RESTOR13
//  ASSGN    SYS004,cuu
//  TLBL     TAPE1,'BACKUP.FILE1'
//  EXEC     IDCAMS,SIZE=AUTO
    RESTORE  OBJECTS((VMC.KSDS.*          -
             VOLUMES(337001))             -
             STDLABEL(TAPE1)              -
             NOREWIND
//  TLBL     TAPE2,'BACKUP.FILE2'
//  EXEC     IDCAMS,SIZE=AUTO
    RESTORE  OBJECTS(*                    -
             VOLUMES(337001))             -
             CATALOG(VSAM.USER.CATALOG.A) -
             STDLABEL(TAPE2)              -
             NOREWIND
/*
/&
```

Restored Objects:

VMC.KSDS.#1	**specified via VMC.KSDS.*, restored to 337001**
VMC.AIX.#1	alternate index of VMC.KSDS.#1, restored to 338001
VMC.AIX.#2	alternate index of VMC.KSDS.#1, restored to 338001
VMC.PATH.#1	path for VMC.AIX.#1, restored to 338001
VMC.PATH.BCL	path for VMC.KSDS.#1, restored to 338001
VMC.KSDS.#2	specified via VMC.KSDS.*, restored to 337001
VSAM.USER.CATALOG.A	specified explicitly

Note: To restore the AIXes and paths to the new device type, specify VMC.AIX.* and VMC.PATH.* as *entrynames*.

Appendix A. Interpreting LISTCAT Output

The options of the LISTCAT command in their combinations tailor the LISTCAT output. This appendix shows the order in which entries are listed and the meanings of the listed fields.

An entry has a *type* (cluster, nonVSAM, and so on) and a *name* by which it is usually listed. Exceptions are:

1. When the ENTRIES parameter is employed. Then, the entries are listed in the order in which they are specified.
2. When the entry types are not restricted by use of one or several options (for example, cluster, space, data). Then, an entry that has *associated* entries is immediately followed by listings of the associated entries. Thus a cluster's data component, and possibly index component, is listed immediately following the cluster. A path is listed immediately following its path entry.

The following:

- Explains the *entry type code* used with field group descriptions.
- Describes the *keyword fields*.
- Discusses *messages* that are issued with LISTCAT jobs.
- Gives *examples* of LISTCAT output.

Entry Type Code Used with Field Group Descriptions

Although the different entry types do not use all of the fields that may be listed (for example, a cluster entry does not have any allocation fields), there is a large set of fields and groupings of fields that are common to most entry types. Accordingly, the fields are discussed as a complete set. Every field or field grouping indicates the entry types to which they apply, as follows:

Code	Meaning
C	Cluster
D	Data
I	Index
G	Alternate Index
R	Path
U	User Catalog
V	Space (that is, a volume entry)
A	NonVSAM

Description of Keyword Fields

The *field names* are in the following groups of related information; the *group names* are:

Allocation Group
Associations Group
Attributes Group
Data Space Group
History Group
NonVSAM Entry, Special Field for
Protection Group
Statistics Group

Volumes Group
Volume Entry, Special Fields for

In the following, the groups are shown in alphabetic order. The field names within every group are shown in alphabetic order (not the order of appearance in the listed entry).

The Allocation Group (D,I)

The fields in this group describe the space allocated to the data or index component defined by the entry.

HALRBA-OR-CI
The highest RBA (plus 1) available within allocated space to store data. For an extra-large dataset, this field contains the high allocated CI.

HUSRBA-OR-CI
The highest RBA (plus 1) within allocated space that actually contains data. This is the highest of the high-used RBAs in the "Volumes Groups", with a possible exception in the case that the file was not successfully closed and has not been verified. For an extra-large dataset, this field contains the high used CI.

SPACE-PRI
Gives the number of units (indicated under TYPE) of space allocated to the data or index component when the cluster was defined. This amount of space is to be allocated whenever a data component (or key range within it, and its associated sequence set, if IMBED is an attribute of the cluster) is extended to a candidate volume.

SPACE-SEC
Gives the number of units (indicated under TYPE) of space to be allocated whenever a file (or key range within it) is extended beyond the primary space on this volume.

SPACE-TYPE
Indicates the unit of space allocation:

BLOCK Blocks
CYLINDER Cylinders
TRACK Tracks
USECLASS-PRI Indicates the primary useclass of a catalog, cluster, or alternate index.
USECLASS-SEC Indicates the secondary useclass of a catalog, cluster, or alternate index.

The Associations Group (G,R,C,D,I)

This grouping lists the types (for example, cluster, data) and entry names of every entry associated with the present entry. A cluster or alternate index entry will indicate its associated path entries and data and index (if a key-sequenced file) entries. Similarly, an index or data entry will indicate its associated cluster or alternate index of which it is a component.

- An alternate-index entry points to:
  - Its associated data and index entries
  - Its base cluster's cluster entry
  - Every associated path entry
- An alternate-index's data entry points to its associated alternate-index entry
- An alternate-index's index entry points to its associated alternate-index entry
- A cluster entry points to:
  - Its associated data entry
  - Every associated path entry
  - For a key-sequenced cluster, its associated index entry
  - For a cluster with alternate indexes, every associated alternate index entry

- A cluster's data entry points to its associated cluster entry
- A cluster's index entry points to its associated cluster entry
- A path entry (that establishes the connection between a base cluster and an alternate index) points to:
  - Its associated alternate index entry, and the alternate index's associated data and index entries
  - The data entry of its associated base cluster
  - For a key-sequenced base cluster, the index entry of its associated base cluster
- Path entry (that establishes the connection to the base cluster) points to:
  - Its associated base cluster entry
  - The data entry of its associated base cluster
  - For a key-sequenced cluster, the index entry of its associated base cluster

Entries are identified as follows:

```
AIX        Identifies an alternate-index entry
CLUSTER    Identifies a cluster entry
DATA       Identifies a data entry
INDEX      Identifies an index entry
NONVSAM    Identifies a nonVSAM file entry
PATH       Identifies a path entry
UCAT       Identifies a user catalog entry
```

The Attributes Group (D,I,G,R)

The fields in this group describe the miscellaneous attributes of the entry. See the DEFINE command for further discussion of most of these attributes.

ACT-DIC-TOKEN
The dictionary token contains the compression status and the dictionary building blocks that constitute the dictionary for a VSE/VSAM file that was defined with the COMPRESSED parameter. Refer to "Working with Compressed Files" in the VSE/VSAM User's Guide and Application Programming.

AVGLRECL
The average length of data records. AVGLRECL equals MAXLRECL when the records are fixed-length.

AXRKP
Indicates, for an AIX, the alternate index Relative Key Position. The offset, from the beginning of the base cluster's data record, at which the alternate-key field begins.

BUFSPACE
The minimum buffer space in virtual storage to be provided by a processing program.

CIFORMAT
For information on *CI format*, refer to "Explicit Define Cluster (Using the DEFINE CLUSTER Command)" in the VSE/VSAM User's Guide and Application Programming.

CI/CA
The number of CIs per CA. (If this field contains 0, the file is a SAM ESDS file. The file is not organized in CA format.)

CISIZE
The size, in bytes, of the entry's control intervals. Refer to the CONTROLINTERVALSIZE parameter. The value listed in CISIZE is the size which either was specified in the parameter, or was computed when the entry was defined. (If this field contains 0, the file is a SAM ESDS file. The file is not organized in CA format.) For more information, refer to "Explicit Define Cluster (Using the DEFINE CLUSTER Command)" in the VSE/VSAM User's Guide and Application Programming.

CMP-ACTIVE
Data compression for the file is active. Refer to "Compression States" in the VSE/VSAM User's Guide and Application Programming.

CMP-PENDING
The file was defined with the COMPRESSED attribute but data compression is not yet active. Refer to "Compression States" in the VSE/VSAM User's Guide and Application Programming.

CMP-REJECT
The file was defined with the COMPRESSED attribute but data compression was rejected. Refer to "Compression States" in the VSE/VSAM User's Guide and Application Programming.

CMP-UNDET
The file was defined with the COMPRESSED attribute but the data compression status could not be determined. There may be a problem with the VSE/VSAM compression control data set. Refer to "Compression States" in the VSE/VSAM User's Guide and Application Programming.

COMPRESS
The file was defined with the COMPRESSED parameter. Refer to "Working with Compressed Files"in the VSE/VSAM User's Guide and Application Programming .

ERASE
Records are to be erased (set to binary 0s) when deleted.

EXCPEXIT
The name of the object's exception exit routine.

EXP-DEFINE
Refer to "Explicit Define Cluster (Using the DEFINE CLUSTER Command)" in the VSE/VSAM User's Guide and Application Programming.

EXTRALARGE
VSAM KSDS file can exceed 4GB in size. CI numbers are displayed instead of RBAs.

IMBED
The sequence-set index record is stored along with its associated data CA.

IMP-DEFINE
Refer to "Implicit Define Cluster" in the VSE/VSAM User's Guide and Application Programming.

INH-UPDATE
The data component cannot be updated. The data component:

- Was exported with INHIBITSOURCE specified, or
- Was exported and imported with INHIBITTARGET specified, or
- Entry was modified by way of ALTER, with INHIBIT specified.

INDEXED
The data component has an index; it is key-sequenced.

INDEX-NUMB
The data component has an index, but is numbered (VRDS).

KEYLEN
The length of the key field in a data record.

MAXLRECL
The maximum length of data or index records. AVGLRECL equals MAXLRECL when the records are fixed-length.

MAXRECS
Identifies the highest-possible relative-record number that can be addressed for this relative-record file. This number depends on the CI size and record size specified for the file. It is calculated independent of the actually allocated space.

NOALLOC
No space is allocated to the file. That is, the file has the potential to have no space; it may have space if it is dynamic.

NOCIFORMAT
Refer to "Explicit Define Cluster (Using the DEFINE CLUSTER Command)" in the VSE/VSAM User's Guide and Application Programming.

NOCOMPRESS
The file was not defined with the COMPRESSED parameter.

NOERASE
Records are not to be erased (set to binary 0s) when deleted.

NOIMBED
The sequence-set index record is not stored along with its associated data CA.

NONINDEXED
The data component has no index; it is entry-sequenced.

NONSPANNED
Data records cannot span CIs.

NONUNIQKEY
For an AIX, more than one data record in the base cluster can contain the same alternate-key value.

NOREPLICAT
Index records are not replicated.

NOREUSE
The file cannot be reused.

NOTUSABLE
The entry is not usable, because space or critical volumes allocated for the file have been deleted by the DELETE FORCE option or by ALTER REMOVEVOLUMES.

NOUPDATE
When the path is opened for processing, its associated base cluster and alternate index are opened but the base cluster's upgrade set is not opened.

NOUPGRADE
The alternate index is not upgraded unless it is opened and being used to access the base cluster's data records.

NOWRITECHK
Write operations are not checked for media recording correctness.

NUMBERED
The cluster is a relative-record file.

ORDERED
Volumes are used for space allocation in the order they were specified when the cluster/alternate index was defined.

RECORDFORMAT
Refer to the "Explicit Define Cluster (Using the DEFINE CLUSTER Command)" in the VSE/VSAM User's Guide and Application Programming.

RECOVERY
A temporary CLOSE is issued as every CA of the file is loaded, so the whole file will not have to be reloaded if a serious error occurs during loading.

RECORDS/CI
Specifies the number of records, or slots, in every CI of a relative-record file.

REPLICATE
Index records are replicated. That is, every index record is repeated on a track of the index's disk device.

REUSE
The file can be reused. That is, its contents are temporary and its high-used RBA can be reset to 0 every time it is opened.

RKP
Relative Key Position, the offset from the beginning of the logical record, for this entry itself, at which the key field begins. Implicitly defined by VSE/VSAM for an alternate index. For a cluster, see the KEYS parameter.

SAMDATASET
The data set was defined with the RECORDFORMAT parameter. Therefore, the data set is usable with the VSE/VSAM Space Management for SAM Function.

For more information, refer to "VSE/VSAM Support for SAM Files" in the VSE/VSAM User's Guide and Application Programming.

SAMLRECL
The logical record size that was specified when the data set was defined. Note: The actual record size may have been overruled by a different value from a DTF specification.

For more information, refer to "Explicit Define Cluster (Using the DEFINE CLUSTER Command)" in the VSE/VSAM User's Guide and Application Programming (refer to *logicalrecordsize*).

SHROPTNS
(n,m) The numbers n and m identify the types of sharing permitted. See SHAREOPTIONS in the DEFINE CLUSTER or DEFINE ALTERNATEINDEX sections for more details.

SPANNED
Data records can be longer than CI length, and can cross, or span, CI boundaries.

SPEED
CLOSE is not issued until the file has been loaded.

SUBALLOC
More than one VSE/VSAM cluster or alternate index can share the data space. A VSE/VSAM catalog might also occupy the data space.

TEMP-EXP
Indicates that the TEMPORARY option was employed when this file was exported (see EXPORT command). This field is omitted should the file not be temporarily exported.

UNIQUE
Only one VSE/VSAM cluster or alternate index can occupy the data space; the cluster or alternate index is unique.

UNIQUEKEY
For an AIX the alternate-key value identifies one, and only one, data record in the base cluster.

UNORDERED
Volumes specified when the cluster was defined can be used for space allocation in any order.

UPDATE
When the path is opened, the upgrade set's alternate indexes (associated with the path's base cluster) are also opened and are updated when the base cluster's contents change.

UPGRADE
When the alternate index's base cluster is opened, the alternate index is also opened and will be updated to reflect any changes to the base cluster's contents.

VSAMDSET
Indicates that this is a VSE/VSAM entry-sequenced file (not a SAM ESDS file).

WRITECHECK
Write operations are checked for correctness.

The Data Space Group (V)

The fields in this group are included by LISTCAT, as part of a volume entry, for every data space on the volume. If a volume contains no data spaces, it is a candidate volume.

ATTRIBUTES
Describes the attributes of the data space.

- **AUTOMATIC** The data space was created by a secondary allocation operation. If a VSE/VSAM file must be given additional space on a volume and all existing data spaces are full, an existing data space will be extended, or a new one allocated on the volume, using catalog DADSM (direct access

device space management). A new data space so allocated is known as an implicit or automatic data space

Note: This attribute is not utilized by z/VSE.

- **CLASS** Indicates the class of space on the storage volume.
- **EXPLICIT** The data space was created explicitly by a:
  - DEFINE MASTERCATALOG or USERCATALOG command, or
  - DEFINE ALTERNATEINDEX or DEFINE CLUSTER command with the UNIQUE attribute specified, or
  - DEFINE SPACE command.

  MASTERCAT The data space contains the master catalog.
  SUBALLOC The data space might contain several VSE/VSAM clusters.
  USERCAT The data space contains a user catalog.
  UNIQUE The data space contains a single (unique) VSE/VSAM file component.

DATASET DIRECTORY
Lists the VSE/VSAM files that can be stored (see CANDIDATE, below) or actually are stored (in whole or in part) in the data space.

- **ATTRIBUTES** Describes the relation between the named file and the data space.

  CANDIDATE The volume is a candidate for storing the file.
  (NULL) The file is stored (in whole or in part) in the data space.
- **DSN** The file or data set name of the object that can be stored or is stored on the volume.
- **EXTENTS** The number of file suballocated extents for the file within the data space.

DATASETS
The number of VSE/VSAM files stored (in whole or in part) in the data space. (The number does not include files for which the volume is a candidate.)

EXTENT-DESCRIPTOR
Describes the data space extent.

BEG-BLOCK
The device address (in decimal) of the extents (FBA devices only).

- **BEG-CCHH** The device address (that is, CC = cylinder and HH = track) of the extent.
- **BLOCKS-TOTAL** The total number of blocks (in decimal) allocated to the data space (FBA devices only).
- **BLOCKS-USED** The number of blocks (in decimal) allocated to files and catalogs (FBA devices only).
- **SPACE-MAP (for Large DASD: BIG-SPC-MAP or FAT-SPC-MAP)**

  For CKD devices: A hexadecimal number that tells what tracks (cylinders for Large DASD) are used and what tracks (cylinders for Large DASD) are free in the extent. The number consists of one or more RLCs (run length codes). The first RLC gives the number of contiguous *used* tracks/cylinders, starting at the beginning of the extent; if all the tracks/cylinders in the extent are used, there is only one RLC. The second RLC gives the number of contiguous *free* tracks/cylinders, beyond the used tracks/cylinders. A third RLC gives used tracks/cylinders again, a fourth gives free tracks/cylinders, and so on.

  A 1-byte RLC gives the number of tracks/cylinders less than 250; a 3-byte RLC gives the number of tracks/cylinders equal to or greater than 250. That is:

  - If the first byte of an RLC is X‘F9’ (249) or less, it is the only byte of the RLC and gives the number of tracks/cylinders.
  - If the first byte of an RLC is X‘FA’ (250) or more, the byte is followed by two more bytes that give the number of tracks/cylinders. (The first byte means nothing more than to look at the next two bytes.)

For FBA devices: A hexadecimal number that tells what blocks are used and what blocks are free in the extent. Every RLC (run length code) is always four bytes long.

- BIG-SPC-MAP and FAT-SPC-MAP show values in cylinders instead of tracks.
- **TRACKS-TOTAL** The total number of tracks (in decimal) allocated to the data space.
- **TRACKS-USED** The number of tracks (in decimal) allocated to files and catalogs.

EXTENTS
The number of file suballocated extents in the data space.

FORMAT-1-LABEL
Identifies the Format-1 label that describes the data space.

- **BLOCK** The address (in decimal) of the Format-1 label in the VTOC (FBA devices only).
- **CCHHR** The device address (that is, CC = cylinder, HH = track, and R = record number) of the Format-1 label in the Volume Table of Contents.
- **TIMESTAMP** The time the data space was allocated (time-of-day clock value and corresponding Julian date (yy.ddd) and time).
- **SEC-ALLOC** Gives the number of units (indicated under TYPE) of space to be allocated whenever the data space is extended.
- **TYPE** Indicates the unit of space allocation:

  BLOCK Blocks
  CYLINDER Cylinders
  TRACK Tracks

The History Group (C,D,G,I,R)

The fields in this group identify the object's owner, identify its release level, and give its creation and expiration dates.

entryname
The name of the cataloged object. The entryname can be specified with the ENTRIES parameter of LISTCAT to identify a catalog entry.

HISTORY
This field includes the following fields:

- **CREATION** The Julian date (yy.ddd) the entry was created.
- **EXPIRATION** The Julian date (yy.ddd) on which the entry can be deleted without specifying the PURGE parameter in the DELETE command. Julian date 99.999 indicates that PURGE is always required to delete the object.
- **OWNER-IDENT** The identity of the owner of the object described by the entry.
- **RELEASE** The release of VSE/VSAM under which the entry was created (not the same as the release number of OS/VS2):

  1 = DOS/VS 28, 29, and 30
  2 = DOS/VS 31 to DOS/VS 34, VSE, VSE/ESA, and z/VSE

The NonVSAM Entry, Special Field For (A)

The special field for a nonVSAM file describes a nonVSAM file stored on tape.

FSEQN
The sequence number (for the tape volume indicated under the "VOLUMES group" keyword VOLSER) of the file on which the nonVSAM file is stored.

The Protection Group (C,D,G,I,R)

The fields in this group describe how the alternate index, cluster, data component, index component, or path defined by the entry is protected. NULL or SUPPRESSED might be listed under PROTECTION:

NULL
Indicates that the object defined by the entry has no passwords.

SUPP
Indicates that the master password of neither the catalog nor the entry was specified, so authority to list protection information is not granted.

ATTEMPTS
Gives the number of times (zero through seven) the console operator is allowed to attempt to enter a correct password.

CODE
Gives the one to eight character code used to tell the console operator what alternate index, catalog, cluster, path, data component, or index component requires him to enter a password. NULL is listed under CODE if a code is not used; the object requiring the password is identified with its full name.

CONTROLPW
The CI password (that is, the password for CI access). NULL indicates no CI password.

MASTERPW
The master password.

READPW
The read-only password. NULL indicates no read-only password.

UPDATEPW
The update password. NULL indicates no update password.

USAR
The contents (1 to 255 bytes, in character format) of the USAR (user-security-authorization record). This is the information specified in the *string* subparameter of the AUTH subparameter of the DEFINE command.

USVR
The name of the user-written program which is to be invoked to verify authorization of any access to the entry; known as the USVR-User Security Verification Routine.

The Statistics Group (D,I)

The fields in this group give numbers and percentages that tell how much VSE/VSAM activity has taken place in the processing of a data or index component.

Note:

1. Activity during managed-SAM access is not recorded in the catalog.
2. The following fields are updated when the file is successfully closed and providing the catalog resides *not* on a read-only device: record fields (REC), split fields (SPLITS), and EXCPs.
3. The statistics group will be zero when the CLOSE disposition of the file results in the resetting or deallocation of the file.
4. Statistics may be incorrect if:
   - Any previous CLOSE did not complete successfully.
   - The file has been accessed through DTF access (managed-SAM access).
   - Control interval access (MACRF=CNV) was used.

   Incorrect statistics remain incorrect until the cluster is re-defined.

FREESPACE-%CI
Percentage of space to be left free in a CI for subsequent processing. This is the value you requested; the actual amount set aside by VSE/VSAM may be different. (Refer to FREESPACE parameter description.)

FREESPACE-%CA
Percentage of CIs to be left free in a CA for subsequent processing. This is the value you requested; the actual amount set aside by VSE/VSAM may be different. (Refer to FREESPACE parameter description).

FREESPACE
For the data component of a KSDS or RRDS: actual number of bytes of unused CAs in the total amount of allocated space. This value is the difference between the high-allocated RBA and the high-used RBA. For a key range file, it is the sum of these differences for every key range. It does not include:

- Free CIs within partially filled CAs.
- Free bytes within partially filled CIs.

For the data component of an extra-large dataset: actual number of unused CIs in the total amount of allocated space.

For the data component of an ESDS: actual number of bytes of unused CIs at the end of the file, based on the total amount of allocated space.

For the index component: actual number of bytes of unused control intervals at the end of the component, based on the total amount of allocated space.

INDEX
This field appears only in an index entry. The fields under it describe activity in the index component.

ENTRIES/SECT The desired number of entries in every section of entries in an index record. Used by VSE/VSAM for further calculations.
HI-LEVEL-RBA The RBA (relative byte address) of the highest-level index record.
LEVELS The number of levels of records in the index. The number is 0 if no records have been loaded into the key-sequenced file to which the index belongs.
SEQ-SET-RBA A decimal field which contains the RBA of the first sequence-set record. The sequence set might be separated from the index set by some quantity of RBA space.

EXCPS
The number of EXCP (execute channel program—SVC 0) macro instructions issued by VSE/VSAM against the data or index component.

EXTENTS
Extents in the data or index component.

REC-DELETED
The number of records that have been deleted from the data or index component.

REC-INSERTED
The number of records inserted into the data component that have not been added to the end of the file.

REC-RETRIEVED
The number of records that have been retrieved from the data or index component, whether for update or not for update.

REC-TOTAL
The total number of records actually in the data or index component. Note that for relative record files this is the number of slots that have been formatted.

REC-UPDATED
The number of records that have been retrieved for update and rewritten. This value does not reflect those records which were deleted. A record that is updated and then deleted is still counted in the update statistics.

SPLITS-CA
The number of control-area splits. Half the data records in a CA were written into a new CA and then were deleted from the old CA.

SPLITS-CI
The number of control-interval splits. Half the data records in a control interval were written into a new CI and then were deleted from the old CI.

SYSTEM-TIMESTAMP
The time (time-of-day clock value and corresponding Julian date (yy.ddd) and time) when the data or index component was last closed after it was opened for an operation that might have changed its contents.

The Volumes Group (D,I)

The fields in this group identify the volume(s) on which a data component, index component, user catalog, or nonVSAM file is stored. It also identifies candidate volume(s) for a data or index component. The fields describe the type of volume and give, for a data or index component, information about the space the object uses on the volume.

- If an entry-sequenced or relative-record cluster's data component has more than one VOLUMES group, every group describes the extents that contain data records for the cluster on a specific volume.
- If a key-sequenced cluster's data component has more than one VOLUMES group, every group describes the extents that contain data records for the cluster, or one of its key ranges, on a specific volume.
- If a key-sequenced cluster's index component has more than one VOLUMES group, every group describes the extents that contain index records for the cluster, or one of its key ranges, on a specific volume. The first VOLUMES group describes the extent that contains the high-level index records (that is, index records in levels above the sequence set level). Each of the next groups describe the extents that contain sequence-set index records for the cluster, or one of its key ranges, on a specific volume. The index component of a key-sequence file with the imbed attribute will have a minimum of two VOLUMES groups, one for the imbedded sequence set and one for a high-level index. The extents for the imbedded sequence set will be the same as those for the data component.

BLOCKS/CA
The number of blocks (in decimal) which comprise a CA. (Applies to FBA devices only.)

BLKS/MIN-CA
The number of blocks (in decimal) that VSE/VSAM can write in a minimum CA unit on the volume. (Applies to FBA devices only.)

DEVTYPE
The type of device to which the volume belongs. For Large DASD, the entry is prefixed either by "BIG-" or "FAT-".

EXTENT-NUMBER
The number of extents allocated for the data or index component on the volume.

EXTENT-TYPE
The type of extents:

00 The extents are contiguous.
40 The extents are not preformatted. The extents can be contiguous.
80 A sequence set occupies a track adjacent to a CA.

EXTENTS
Gives the physical and relative-byte addresses of every extent. If extent-number is 0 (NOALLOC), these fields that follow (BLOCKS to TRACKS) are not printed.

BLOCKS
The number of blocks (in decimal) in the extent. (Applies to FBA devices only.)

HIGH-BLOCK
The device address of the end of the extent. (Applies to FBA devices only.)

HIGH-CCHH
The device address (that is, CC = cylinder and HH = track) of the end of the extent.

HI-RBA-OR-CI
A decimal field containing the RBA (relative byte address) of the end of the extent. For an extra-large dataset, this field contains the last CI in the extent.

LOW-BLOCK
The device address of the beginning of the extent. (Applies to FBA devices only).

LOW-CCHH
The device address (that is, CC = cylinder and HH = track) of the beginning of the extent.

LOW-RBA-OR-CI
A decimal field containing the RBA (relative byte address) of the beginning of the extent. For an extra-large dataset, this field contains the first CI in the extent.

TRACKS | CYLINDERS
The size of the extent, from low to high device addresses. The extent is given in CYLINDERS for Large DASD, otherwise in TRACKS.

HIGH-KEY
For a key-sequenced file with the KEYRANGE attribute, the highest hexadecimal value allowed in the key field of a record in the key range. A maximum of 64 bytes can appear in HIGH-KEY.

Multiple key ranges might reside on a single volume; the volumes group is repeated for every such key range field.

HI-KEY-RBA
For a key-sequenced file, the RBA (relative byte address) in decimal of the CI on the volume that contains the highest-keyed record in the file or key range.

Multiple key ranges might reside on a single volume; the volumes group is repeated for every such key range field.

LOW-KEY
For a key-sequenced file with the KEYRANGE attribute, the lowest hexadecimal value allowed in the key field of a record in the file or key range. A maximum of 64 bytes can appear in LOW-KEY.

Multiple key ranges might reside on a single volume; the volumes group is repeated for every such key range field.

PHYRECS/TRK
The number of blocks (physical records of the size indicated under PHYRECS-SIZE) that VSE/VSAM can write on a track on the volume.

Multiple key ranges might reside on a single volume; the volumes group is repeated for every such key range field.

PHYREC-SIZE
The number of bytes that VSE/VSAM uses for a physical block (physical record) in the data or index component.

HALRBA-OR-CI
The highest RBA (plus 1) available within allocated space to store data component, its key-range, the index component, or the sequence set records of a key range. For an extra-large dataset, this field contains the highest available CI.

HUSRBA-OR-CI
The highest RBA or CI as follows:

For the data component of a KSDS or RRDS: The highest RBA of a CA (plus 1) within space allocated to the component (or to a key range) that actually contains data.
For the data component of an extra-large dataset: The highest CI (plus 1) that actually contains data.
For the data component of an ESDS: The highest RBA of a CI (plus 1) that actually contains data.

For the index component: The highest RBA of a CI (plus 1) within space allocated to the component (or to sequence set records of a key range) that actually contains index records.

The HUSRBA-OR-CI entry in the candidate volume entry is not always connected to the volume entry itself. It shows the high-used-RBA of the (index or data) component, or the CI for an extra-large dataset.

TRACKS/CA
The number of tracks which comprise a CA. (This value is computed when the entry is defined, and reflects the optimum size of the CA for the given device type and the nature of the entry -- whether indexed, non-indexed, or numbered.) For a key-sequenced file with the imbedded attribute, this value includes the sequence set track.

VOLFLAG
Indicates whether the volume is a candidate volume or the first or a subsequent volume on which data in a given key range is stored.

CANDIDATE
The volume is a candidate for storing the data or index component, but which as yet has no space allocated upon it for the entry.

OVERFLOW
The volume is an overflow volume on which data records in a key range are stored. The key range begins on another (a PRIME) volume.

PRIME
The volume is the first volume on which data records in a key range are stored.

VOLSER
The one through six character volume serial.

The Volume Entry, Special Fields For (V)

The special fields for a volume entry describe the characteristics of the space that VSE/VSAM uses on the volume.

volume serial number
The name of the cataloged volume entry. The volume serial number can be specified in the ENTRIES parameter of LISTCAT to identify the volume entry.

BLKS/MAX-CA
The number of blocks (in decimal) in the largest CA on the volume. (Applies to FBA devices only.)

BLKS/MIN-CA
The number of blocks (in decimal) that VSE/VSAM can use in a minimum CA unit on the volume. (Applies to FBA devices only.)

BLOCKS/VOL
The number of blocks (in decimal) that VSE/VSAM can use on the volume. This does not include alternate blocks reserved for error recovery. (Applies to FBA devices only.)

BYTES/TRK
The number of bytes that VSE/VSAM can use on every track on the volume, including alternate track cylinders.

CYLS/VOL
The number of cylinders that VSE/VSAM can use on the volume, including alternate track cylinders.

DATASETS-ON-VOL
The number of VSE/VSAM clusters that reside, in whole or in part, on the volume. If the number of data spaces is zero, then this is a candidate volume only and the DATASPACE grouping is omitted. (The number includes files for which the volume is a candidate.)

DATASPCS-ON-VOL
The number of VSE/VSAM data spaces on the volume. If the number of data spaces is zero, then this is a candidate volume only and the DATASPACE grouping is omitted.

DEVTYPE
The type of device to which the volume belongs. For Large DASD, the entry is prefixed either by "BIG-" or "FAT-".

MAX-PHYREC-SZ
The size of the largest block (physical record) that VSE/VSAM can write on the volume.

MAX-EXT/ALLOC
Set to 5, which is the maximum number of noncontiguous extents permitted in one primary or secondary allocation.

TRKS/CYL
The number of tracks in every cylinder on the volume.

VOLUME-TIMESTAMP
The time (time-of-day clock value and corresponding Julian date, dd.yyy, and time) when VSE/VSAM last changed the contents of the volume. The Format-4 label contains the time stamp at offset 76 (X'4C').

LISTCAT and IDCAMS Output Messages

When the LISTCAT job completes, IDCAMS provides messages and diagnostic information.

For explanations to error messages, refer to "IDCAMS Messages and Codes" in z/VSE Messages and Codes Volume 2.

When your LISTCAT job completes *successfully*, IDCAMS provides messages at the end of the *Output Listings* (as shown, for example, in Figure 23 on page 285). In detail, the messages are shown in Figure 22 on page 284; their meaning is as follows:

- The message line "LISTING FROM ..." identifies the catalog (here MJKCAT) which contains the listed entries.
- The next group of lines (starting with AIX) specifies the number of every entry-type and the total number of entries, that were listed. This statistical information can help you determine the approximate size of your catalog (in number of records).
- The message line "THE NUMBER OF ..." shows the number of entries that could not be listed because the appropriate password was not specified.
- The last message line indicates that the LISTCAT command (FUNCTION) and the job step (IDCAMS) completed successfully.

```
      LISTING FROM CATALOG -- MJKCAT

   THE NUMBER OF ENTRIES PROCESSED WAS:
        AIX ------------------1
        CLUSTER ---------------2
        DATA ------------------3
        INDEX -----------------3
        NONVSAM ---------------1
        PATH ------------------1
        SPACE -----------------2
        USERCATALOG -----------0
        TOTAL ----------------13

   THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0

   IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

Figure 22. Example of Messages That Follow the Entries of Output Listings

LISTCAT Output Listing

When you specify LISTCAT without any parameters, the entryname and type of every entry is listed (Figure 23 on page 285). The same listing would result if the NAMES parameter were specified.

You can use this type of listing to list the name of every cataloged object and to determine the number of entries in the catalog. The total number of entries is an approximate size, in records, of your catalog.

```
// EXEC IDCAMS,SIZE=AUTO                                              LIS00040

IDCAMS  SYSTEM SERVICES                                          TIME: 10:41:20
02/21/1997   PAGE   1

   LISTCAT CATALOG(TST.UCAT)                                          LIS00050
IDCAMS  SYSTEM SERVICES                                          TIME: 10:41:20
02/21/1997   PAGE   2
                            LISTING FROM CATALOG -- TST.UCAT
VOLUME -------- CTS240
VOLUME -------- CTS400
CLUSTER ------- IMP.SAM.ESDS
   DATA ------- TC27AB6A.VSAMDSET.DFD97051.TAE3FF4D.TC27AB6A
AIX ----------- KSDS1.AIX
   DATA ------- KSDS1.AIX.DATA
   INDEX ------ KSDS1.AIX.INDEX
   PATH ------- KSDS1.PATH1
CLUSTER ------- TST.ESDS1
   DATA ------- TST.ESDS1.@D@
CLUSTER ------- TST.ESDS2
   DATA ------- TST.ESDS2.@D@
CLUSTER ------- TST.KSDS1
   DATA ------- TST.KSDS1.@D@
   INDEX ------ TST.KSDS1.@I@
CLUSTER ------- TST.KSDS2
   DATA ------- TST.KSDS2.@D@
   INDEX ------ TST.KSDS2.@I@
CLUSTER ------- TST.RRDS1
   DATA ------- TST.RRDS1.@D@
CLUSTER ------- TST.UCAT
   DATA ------- VSAM.CATALOG.BASE.DATA.RECORD
   INDEX ------ VSAM.CATALOG.BASE.INDEX.RECORD
CLUSTER ------- VSAM.COMPRESS.CONTROL
IDCAMS  SYSTEM SERVICES                                          TIME: 10:41:20
02/21/1997   PAGE   3
                            LISTING FROM CATALOG -- TST.UCAT
   DATA ------- VSAM.COMPRESS.CONTROL.@D@
   INDEX ------ VSAM.COMPRESS.CONTROL.@I@
IDCAMS  SYSTEM SERVICES                                          TIME: 10:41:20
02/21/1997   PAGE   4
                            LISTING FROM CATALOG -- TST.UCAT
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX ------------------1
                   CLUSTER --------------8
                   DATA -----------------9
                   INDEX ----------------5
                   NONVSAM --------------0
                   PATH -----------------1
                   SPACE ----------------2
                   USERCATALOG ----------0
                   TOTAL ----------------26
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

Figure 23. Example of LISTCAT Output When No Parameters are Specified

LISTCAT VOLUME Output Listing

When the LISTCAT command is specified with the VOLUME parameter, the volume serial number and device type of every volume that contains part or all of the cataloged object are listed (Figure 24 on page 285).

Figure 24. Example of LISTCAT VOLUME Output

```
// EXEC IDCAMS,SIZE=AUTO                                              LIS00040

IDCAMS  SYSTEM SERVICES                                          TIME: 10:44:53
02/21/1997   PAGE   1

   LISTCAT VOLUME CATALOG(TST.UCAT)                                   LIS00050
IDCAMS  SYSTEM SERVICES                                          TIME: 10:44:53
02/21/1997   PAGE   2
                            LISTING FROM CATALOG -- TST.UCAT
VOLUME -------- CTS240
     HISTORY
```

```
       RELEASE----------------2
     VOLUMES
       VOLSER------------CTS240     DEVTYPE------------3380
VOLUME -------- CTS400
     HISTORY
       RELEASE----------------2
     VOLUMES
       VOLSER------------CTS400     DEVTYPE------------3380
CLUSTER ------- IMP.SAM.ESDS
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2005.243
   DATA ------- TC27AB6A.VSAMDSET.DFD97051.TAE3FF4D.TC27AB6A
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2005.243
     VOLUMES
       VOLSER------------CTS400     DEVTYPE------------3380
AIX ----------- KSDS1.AIX
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
   DATA ------- KSDS1.AIX.DATA
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------CTS240     DEVTYPE------------3380
   INDEX ------ KSDS1.AIX.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------CTS240     DEVTYPE------------3380
   PATH ------- KSDS1.PATH1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
IDCAMS  SYSTEM SERVICES                                          TIME: 10:44:53
02/21/1997   PAGE   3
                            LISTING FROM CATALOG -- TST.UCAT
CLUSTER ------- TST.ESDS1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1997.058
   DATA ------- TST.ESDS1.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1997.058
     VOLUMES
       VOLSER------------CTS240     DEVTYPE------------3380
CLUSTER ------- TST.ESDS2
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.365
   DATA ------- TST.ESDS2.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.365
     VOLUMES
       VOLSER------------CTS400     DEVTYPE------------3380
CLUSTER ------- TST.KSDS1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2067.295
   DATA ------- TST.KSDS1.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2067.295
     VOLUMES
       VOLSER------------CTS240     DEVTYPE------------3380
   INDEX ------ TST.KSDS1.@I@
```

```
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2067.295
     VOLUMES
       VOLSER------------CTS240     DEVTYPE-------------3380
CLUSTER ------- TST.KSDS2
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.052
       RELEASE----------------2     EXPIRATION------0000.000
IDCAMS  SYSTEM SERVICES                                          TIME: 10:44:53
02/21/1997   PAGE   4
                            LISTING FROM CATALOG -- TST.UCAT
   DATA ------- TST.KSDS2.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.052
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------CTS400     DEVTYPE-------------3380
   INDEX ------ TST.KSDS2.@I@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.052
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------CTS400     DEVTYPE-------------3380
CLUSTER ------- TST.RRDS1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.366
   DATA ------- TST.RRDS1.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.366
     VOLUMES
       VOLSER------------CTS240     DEVTYPE-------------3380
CLUSTER ------- TST.UCAT
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
   DATA ------- VSAM.CATALOG.BASE.DATA.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------CTS240     DEVTYPE-------------3380
   INDEX ------ VSAM.CATALOG.BASE.INDEX.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------CTS240     DEVTYPE-------------3380
CLUSTER ------- VSAM.COMPRESS.CONTROL
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
IDCAMS  SYSTEM SERVICES                                          TIME: 10:44:53
02/21/1997   PAGE   5
                            LISTING FROM CATALOG -- TST.UCAT
       RELEASE----------------2     EXPIRATION------0000.000
   DATA ------- VSAM.COMPRESS.CONTROL.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------CTS240     DEVTYPE-------------3380
   INDEX ------ VSAM.COMPRESS.CONTROL.@I@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------CTS240     DEVTYPE-------------3380
IDCAMS  SYSTEM SERVICES                                          TIME: 10:44:53
02/21/1997   PAGE   6
                            LISTING FROM CATALOG -- TST.UCAT
```

```
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX ------------------1
                   CLUSTER --------------8
                   DATA -----------------9
                   INDEX ----------------5
                   NONVSAM --------------0
                   PATH -----------------1
                   SPACE ----------------2
                   USERCATALOG ----------0
                   TOTAL ----------------26
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

LISTCAT SPACE ALL Output Listing

When the LISTCAT command is specified with the SPACE and ALL parameters, all the information for every volume entry in the catalog is listed (Figure 25 on page 288).

You can use this type of listing to determine how much space on every cataloged volume is allocated to VSE/VSAM data spaces. You may have to list the volume's table of contents (VTOC) to determine how all of the volume's space is allocated.

Figure 25. Example of LISTCAT SPACE ALL Output

```
// EXEC IDCAMS,SIZE=AUTO                                                LIS00040

IDCAMS  SYSTEM SERVICES                                  TIME: 10:45:48        02/21/1997
PAGE   1

   LISTCAT SPACE ALL CATALOG(TST.UCAT)                                  LIS00050
IDCAMS  SYSTEM SERVICES                                  TIME: 10:45:48        02/21/1997
PAGE   2
                          LISTING FROM CATALOG -- TST.UCAT
VOLUME -------- CTS240
     HISTORY
       RELEASE----------------2
     CHARACTERISTICS
       BYTES/TRK----------47968     DEVTYPE------------3380     MAX-PHYREC-SZ------47476     DATASETS-
ON-VOL--------9
       TRKS/CYL-------------15     VOLUME-TIMESTAMP:           MAX-EXT/ALLOC----------5     DATASPCS-
ON-VOL--------2
       CYLS/VOL------------140          1997.051   13:52:04
                                        X'AE3FE7CF372A6005'
     DATASPACE
       DATASETS---------------2     FORMAT-1-LABEL:             ATTRIBUTES:
       EXTENTS----------------1     CCHHR------X'0000000104'    SUBALLOC
       SEC-ALLOC--------------0     TIMESTAMP                   EXPLICIT
       TYPE---------------TRACK          1997.051   13:50:20    USERCAT
       CLASS------------------0          X'AE3FE76B86B78C05'
       EXTENT-DESCRIPTOR:
       TRACKS-TOTAL----------10     BEG-CCHH-----X'00000002'    SPACE-MAP-----------0A
       TRACKS-USED-----------10
       DATASET-DIRECTORY:
         DSN----TST.UCAT                                        ATTRIBUTES--------(NULL)
EXTENTS----------------3
         DSN----VSAM.COMPRESS.CONTROL.@D@                       ATTRIBUTES--------(NULL)
EXTENTS----------------1
     DATASPACE
       DATASETS---------------7     FORMAT-1-LABEL:             ATTRIBUTES:
       EXTENTS----------------1     CCHHR------X'0000000105'    SUBALLOC
       SEC-ALLOC--------------1     TIMESTAMP                   EXPLICIT
       TYPE------------CYLINDER          1997.051   13:52:04
       CLASS------------------0          X'AE3FE7CF372A6005'
       EXTENT-DESCRIPTOR:
       TRACKS-TOTAL----------30     BEG-CCHH-----X'00470000'    SPACE-MAP-----------1806
       TRACKS-USED-----------24
       DATASET-DIRECTORY:
         DSN----VSAM.COMPRESS.CONTROL.@I@                       ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----TST.ESDS1.@D@                                   ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----TST.KSDS1.@D@                                   ATTRIBUTES--------(NULL)
EXTENTS----------------6
         DSN----TST.KSDS1.@I@                                   ATTRIBUTES--------(NULL)
EXTENTS----------------1
```

```
         DSN----TST.RRDS1.@D@                                    ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----KSDS1.AIX.DATA                                   ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----KSDS1.AIX.INDEX                                  ATTRIBUTES--------(NULL)
EXTENTS----------------1
VOLUME --------- CTS400
     HISTORY
       RELEASE----------------2
     CHARACTERISTICS
       BYTES/TRK----------47968     DEVTYPE------------3380     MAX-PHYREC-SZ------47476     DATASETS-
ON-VOL--------4
       TRKS/CYL-------------15     VOLUME-TIMESTAMP:           MAX-EXT/ALLOC----------5     DATASPCS-
ON-VOL--------1
       CYLS/VOL-------------50          1997.051   13:52:33
                                        X'AE3FE7EAE6120C05'
     DATASPACE
       DATASETS-------------4     FORMAT-1-LABEL:             ATTRIBUTES:
IDCAMS  SYSTEM SERVICES                                         TIME: 10:45:48        02/21/1997
PAGE   3
                             LISTING FROM CATALOG -- TST.UCAT
       EXTENTS----------------1     CCHHR------X'0031000E03'     SUBALLOC
       SEC-ALLOC--------------1     TIMESTAMP                    EXPLICIT
       TYPE------------CYLINDER          1997.051   13:52:33
       CLASS-------------------0          X'AE3FE7EAE6120C05'
       EXTENT-DESCRIPTOR:
       TRACKS-TOTAL----------30     BEG-CCHH-----X'00010000'     SPACE-MAP-----------1E
       TRACKS-USED-----------30
       DATASET-DIRECTORY:
         DSN----TST.ESDS2.@D@                                  ATTRIBUTES--------(NULL)
EXTENTS----------------4
         DSN----TST.KSDS2.@D@                                  ATTRIBUTES--------(NULL)
EXTENTS---------------10
         DSN----TST.KSDS2.@I@                                  ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----TC27AB6A.VSAMDSET.DFD97051.TAE3FF4D.TC27AB6A   ATTRIBUTES--------(NULL)
EXTENTS----------------1
IDCAMS  SYSTEM SERVICES                                         TIME: 10:45:48        02/21/1997
PAGE   4
                             LISTING FROM CATALOG -- TST.UCAT
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------0
                   CLUSTER ---------------0
                   DATA ------------------0
                   INDEX -----------------0
                   NONVSAM ---------------0
                   PATH ------------------0
                   SPACE -----------------2
                   USERCATALOG -----------0
                   TOTAL -----------------2
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

LISTCAT ALL Output Listing

When you specify the LISTCAT command and include the ALL parameter, all the information for every catalog entry is listed (Figure 26 on page 290). This example illustrates the LISTCAT output for every type of catalog entry.

You can use this type of listing to obtain all cataloged information (except password and security information) about every entry that is listed. When you want to list an entry's passwords, you must provide the catalog's master password (which results in listing the passwords of every password-protected entry) or every entry's master password.

Note: When ENTRIES is specified, you specify only those entry names that identify catalog entries which are not volume entries. If a volume serial number is specified with the ENTRIES parameter, then entry names of other entry types cannot also be specified. However, if the ENTRIES parameter is not specified and if entry types are not specified (that is, CLUSTER, SPACE, DATA, and so on), all entries in the catalog, including volume entries, are listed.

Figure 26. Example of LISTCAT ALL Output

```
// EXEC IDCAMS,SIZE=AUTO                                                LIS00040

IDCAMS  SYSTEM SERVICES                                    TIME: 10:03:24        02/21/1997
PAGE   1

   LISTCAT ALL CATALOG(TST.UCAT)                                        LIS00050
IDCAMS  SYSTEM SERVICES                                    TIME: 10:03:24        02/21/1997
PAGE   2
                          LISTING FROM CATALOG -- TST.UCAT
VOLUME -------- CTS240
     HISTORY
       RELEASE----------------2
     CHARACTERISTICS
       BYTES/TRK----------47968     DEVTYPE-------------3380     MAX-PHYREC-SZ------47476     DATASETS-
ON-VOL--------9
       TRKS/CYL-----------15     VOLUME-TIMESTAMP:            MAX-EXT/ALLOC----------5     DATASPCS-
ON-VOL--------2
       CYLS/VOL-----------140          1997.051   13:52:04
                                       X'AE3FE7CF372A6005'
     DATASPACE
       DATASETS-----------2     FORMAT-1-LABEL:              ATTRIBUTES:
       EXTENTS------------1     CCHHR------X'0000000104'     SUBALLOC
       SEC-ALLOC----------0     TIMESTAMP                    EXPLICIT
       TYPE---------------TRACK         1997.051   13:50:20     USERCAT
       CLASS--------------0             X'AE3FE76B86B78C05'
       EXTENT-DESCRIPTOR:
       TRACKS-TOTAL-------10     BEG-CCHH-----X'00000002'     SPACE-MAP-----------0A
       TRACKS-USED--------10
       DATASET-DIRECTORY:
         DSN----TST.UCAT                                      ATTRIBUTES--------(NULL)
EXTENTS----------------3
         DSN----VSAM.COMPRESS.CONTROL.@D@                     ATTRIBUTES--------(NULL)
EXTENTS----------------1
     DATASPACE
       DATASETS-----------7     FORMAT-1-LABEL:              ATTRIBUTES:
       EXTENTS------------1     CCHHR------X'0000000105'     SUBALLOC
       SEC-ALLOC----------1     TIMESTAMP                    EXPLICIT
       TYPE------------CYLINDER         1997.051   13:52:04
       CLASS--------------0             X'AE3FE7CF372A6005'
       EXTENT-DESCRIPTOR:
       TRACKS-TOTAL-------30     BEG-CCHH-----X'00470000'     SPACE-MAP-----------1806
       TRACKS-USED--------24
       DATASET-DIRECTORY:
         DSN----VSAM.COMPRESS.CONTROL.@I@                     ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----TST.ESDS1.@D@                                 ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----TST.KSDS1.@D@                                 ATTRIBUTES--------(NULL)
EXTENTS----------------6
         DSN----TST.KSDS1.@I@                                 ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----TST.RRDS1.@D@                                 ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----KSDS1.AIX.DATA                                ATTRIBUTES--------(NULL)
EXTENTS----------------1
         DSN----KSDS1.AIX.INDEX                               ATTRIBUTES--------(NULL)
EXTENTS----------------1
VOLUME -------- CTS400
     HISTORY
       RELEASE----------------2
     CHARACTERISTICS
       BYTES/TRK----------47968     DEVTYPE-------------3380     MAX-PHYREC-SZ------47476     DATASETS-
ON-VOL--------4
       TRKS/CYL-----------15     VOLUME-TIMESTAMP:            MAX-EXT/ALLOC----------5     DATASPCS-
ON-VOL--------1
       CYLS/VOL-----------50           1997.051   13:52:33
                                       X'AE3FE7EAE6120C05'
     DATASPACE
       DATASETS-----------4     FORMAT-1-LABEL:              ATTRIBUTES:
IDCAMS  SYSTEM SERVICES                                    TIME: 10:03:24        02/21/1997
PAGE   3
                          LISTING FROM CATALOG -- TST.UCAT
       EXTENTS------------1     CCHHR------X'0031000E03'     SUBALLOC
       SEC-ALLOC----------1     TIMESTAMP                    EXPLICIT
       TYPE------------CYLINDER         1997.051   13:52:33
       CLASS--------------0             X'AE3FE7EAE6120C05'
       EXTENT-DESCRIPTOR:
       TRACKS-TOTAL-------30     BEG-CCHH-----X'00010000'     SPACE-MAP-----------1E
```

```
       TRACKS-USED-----------30
       DATASET-DIRECTORY:
         DSN----TST.ESDS2.@D@                                    ATTRIBUTES--------(NULL)
EXTENTS---------------4
         DSN----TST.KSDS2.@D@                                    ATTRIBUTES--------(NULL)
EXTENTS--------------10
         DSN----TST.KSDS2.@I@                                    ATTRIBUTES--------(NULL)
EXTENTS---------------1
         DSN----TC27AB6A.VSAMDSET.DFD97051.TAE3FF4D.TC27AB6A     ATTRIBUTES--------(NULL)
EXTENTS---------------1
CLUSTER ------- IMP.SAM.ESDS
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2005.243
     PROTECTION----------(NULL)
     ASSOCIATIONS
       DATA-----TC27AB6A.VSAMDSET.DFD97051.TAE3FF4D.TC27AB6A
   DATA ------- TC27AB6A.VSAMDSET.DFD97051.TAE3FF4D.TC27AB6A
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2005.243
     PROTECTION----------(NULL)
     ASSOCIATIONS
       CLUSTER--IMP.SAM.ESDS
     ATTRIBUTES
       KEYLEN-----------------0     AVGLRECL-------------80     BUFSPACE------------4096
CISIZE--------------2048
       RKP--------------------0     MAXLRECL-------------80     EXCPEXIT----------(NULL)     CI/
CA-----------------0
       SAMLRECL-------------80     RECFORMAT-------FIXUNBLK
       SHROPTNS(1,3)      SPEED     SUBALLOC         NOERASE     NOCOMPRESS    NONINDEXED
SAMDATASET    IMP-DEFINE
       NOWRITECHK       NOIMBED     NOREPLICAT     UNORDERED     REUSE         NONSPANNED
     STATISTICS
       REC-TOTAL--------------0     SPLITS-CI--------------0     EXCPS------------------0
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------0          1997.051   14:50:28
       REC-RETRIEVED----------0     FREESPACE----------34816          X'AE3FF4DCA308BE01'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------36864
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI--------2048
     VOLUME
       VOLSER------------CTS400     PHYREC-SIZE---------2048     HALRBA-OR-CI-------36864     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------18     HUSRBA-OR-CI--------2048     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE   4
                            LISTING FROM CATALOG -- TST.UCAT
       EXTENTS:
       LOW-CCHH-----X'00010003'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00010003'     HI-RBA-OR-CI-------36863
AIX ----------- KSDS1.AIX
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       DATA-----KSDS1.AIX.DATA
       INDEX----KSDS1.AIX.INDEX
       CLUSTER--TST.KSDS1
       PATH-----KSDS1.PATH1
     ATTRIBUTES
       UPGRADE
   DATA ------- KSDS1.AIX.DATA
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       AIX------KSDS1.AIX
     ATTRIBUTES
       KEYLEN-----------------7     AVGLRECL------------4086     BUFSPACE------------8704
CISIZE--------------4096
       RKP--------------------5     MAXLRECL-----------32600     EXCPEXIT----------(NULL)     CI/
CA-----------------10
       AXRKP------------------3
       SHROPTNS(1,3)   RECOVERY     SUBALLOC         NOERASE     NOCOMPRESS       INDEXED
NOWRITECHK       NOIMBED
```

```
       NOREPLICAT     UNORDERED     NOREUSE          SPANNED     NONUNIQKEY
     STATISTICS
       REC-TOTAL------------721     SPLITS-CI--------------0     EXCPS------------------4
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------0          1997.052   10:02:38
       REC-RETRIEVED----------0     FREESPACE--------------0          X'AE40F6642C066001'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------40960
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI-------40960
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE---------4096     HALRBA-OR-CI-------40960     EXTENT-
NUMBER----------1

       DEVTYPE-------------3380     PHYRECS/TRK-----------10     HUSRBA-OR-CI-------40960     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00480002'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00480002'     HI-RBA-OR-CI-------40959
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE   5
                              LISTING FROM CATALOG -- TST.UCAT
   INDEX ------ KSDS1.AIX.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       AIX------KSDS1.AIX
     ATTRIBUTES
       KEYLEN-----------------7     AVGLRECL---------------0     BUFSPACE---------------0
CISIZE---------------512
       RKP--------------------5     MAXLRECL-------------505     EXCPEXIT----------(NULL)     CI/
CA-----------------46
       SHROPTNS(1,3)   RECOVERY     SUBALLOC         NOERASE     NOCOMPRESS    NOWRITECHK
NOIMBED       NOREPLICAT
       UNORDERED        NOREUSE
     STATISTICS
       REC-TOTAL--------------1     SPLITS-CI--------------0     EXCPS------------------4     INDEX:
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
LEVELS-----------------1
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:            ENTRIES/
SECT-----------3
       REC-UPDATED------------0     FREESPACE-%CA----------0          1997.052   10:02:38     SEQ-SET-
RBA------------0
       REC-RETRIEVED----------0     FREESPACE----------23040          X'AE40F66432A27601'     HI-LEVEL-
RBA-----------0
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------23552
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI---------512
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE----------512     HALRBA-OR-CI-------23552     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------46     HUSRBA-OR-CI---------512     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00480003'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00480003'     HI-RBA-OR-CI-------23551
   PATH ------- KSDS1.PATH1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       AIX------KSDS1.AIX
       DATA-----KSDS1.AIX.DATA
       INDEX----KSDS1.AIX.INDEX
       DATA-----TST.KSDS1.@D@
       INDEX----TST.KSDS1.@I@
     ATTRIBUTES
       UPDATE
CLUSTER ------- TST.ESDS1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1997.058
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE   6
                              LISTING FROM CATALOG -- TST.UCAT
```

```
     PROTECTION----------(NULL)
     ASSOCIATIONS
       DATA-----TST.ESDS1.@D@
   DATA ------- TST.ESDS1.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1997.058
     PROTECTION----------(NULL)
     ASSOCIATIONS
       CLUSTER--TST.ESDS1
     ATTRIBUTES
       KEYLEN-----------------0     AVGLRECL-------------256     BUFSPACE------------8192
CISIZE--------------4096
       RKP--------------------0     MAXLRECL------------4086     EXCPEXIT----------(NULL)     CI/
CA-----------------20
       ACT-DIC-TOKEN ----X'80000000000000000000000000000000000000000000000000000000000000000000000000000000'
       SHROPTNS(2,3)   RECOVERY     SUBALLOC          NOERASE     COMPRESSED    CMP-REJECT
NONINDEXED    VSAMDATSET
       NOWRITECHK       NOIMBED     NOREPLICAT     UNORDERED     NOREUSE       NONSPANNED
     STATISTICS
       REC-TOTAL------------331     SPLITS-CI--------------0     EXCPS-----------------42
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------0          1997.051   16:55:54
       REC-RETRIEVED--------331     FREESPACE----------69632          X'AE4010E688333000'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------4     USECLASS-PRI-----------0     HALRBA-OR-CI------163840
       SPACE-SEC--------------2     USECLASS-SEC-----------0     HUSRBA-OR-CI-------94208
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE---------4096     HALRBA-OR-CI------163840     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------10     HUSRBA-OR-CI-------94208     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------2
       EXTENTS:
       LOW-CCHH-----X'00470001'     LOW-RBA-OR-CI----------0     TRACKS-----------------4
       HIGH-CCHH----X'00470004'     HI-RBA-OR-CI------163839
CLUSTER ------- TST.ESDS2
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.365
     PROTECTION----------(NULL)
     ASSOCIATIONS
       DATA-----TST.ESDS2.@D@
   DATA ------- TST.ESDS2.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.365
     PROTECTION----------(NULL)
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE   7
                          LISTING FROM CATALOG -- TST.UCAT
     ASSOCIATIONS
       CLUSTER--TST.ESDS2
     ATTRIBUTES
       KEYLEN-----------------0     AVGLRECL-------------256     BUFSPACE------------8192
CISIZE--------------4096
       RKP--------------------0     MAXLRECL-----------32000     EXCPEXIT----------(NULL)     CI/
CA-----------------20
       ACT-DIC-TOKEN ----X'80000000000000000000000000000000000000000000000000000000000000000000000000000000'
       SHROPTNS(1,3)   RECOVERY     NOALLOC           NOERASE     COMPRESSED    CMP-REJECT
NONINDEXED    VSAMDATSET
       NOWRITECHK       NOIMBED     NOREPLICAT     UNORDERED     REUSE            SPANNED     NEVEREXPIR
     STATISTICS
       REC-TOTAL-----------1200     SPLITS-CI--------------0     EXCPS-----------------93
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------4
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------0          1997.051   17:05:37
       REC-RETRIEVED-------1200     FREESPACE--------------0          X'AE4013122440D200'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------2     USECLASS-PRI-----------0     HALRBA-OR-CI------327680
       SPACE-SEC--------------2     USECLASS-SEC-----------0     HUSRBA-OR-CI------327680
     VOLUME
       VOLSER------------CTS400     PHYREC-SIZE---------4096     HALRBA-OR-CI------327680     EXTENT-
NUMBER----------4
       DEVTYPE-------------3380     PHYRECS/TRK-----------10     HUSRBA-OR-CI------327680     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------2
       EXTENTS:
       LOW-CCHH-----X'00010004'     LOW-RBA-OR-CI----------0     TRACKS-----------------2
```

```
       HIGH-CCHH----X'00010005'     HI-RBA-OR-CI-------81919
       LOW-CCHH-----X'00010006'     LOW-RBA-OR-CI------81920     TRACKS-----------------2
       HIGH-CCHH----X'00010007'     HI-RBA-OR-CI------163839
       LOW-CCHH-----X'00010008'     LOW-RBA-OR-CI-----163840     TRACKS-----------------2
       HIGH-CCHH----X'00010009'     HI-RBA-OR-CI------245759
       LOW-CCHH-----X'0001000A'     LOW-RBA-OR-CI-----245760     TRACKS-----------------2
       HIGH-CCHH----X'0001000B'     HI-RBA-OR-CI------327679
CLUSTER ------- TST.KSDS1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2067.295
     PROTECTION----------(NULL)
     ASSOCIATIONS
       DATA-----TST.KSDS1.@D@
       INDEX----TST.KSDS1.@I@
       AIX------KSDS1.AIX
   DATA ------- TST.KSDS1.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2067.295
     PROTECTION----------(NULL)
     ASSOCIATIONS
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE   8
                            LISTING FROM CATALOG -- TST.UCAT
       CLUSTER--TST.KSDS1
     ATTRIBUTES
       KEYLEN-----------------10     AVGLRECL-------------255     BUFSPACE------------8704
CISIZE--------------4096
       RKP--------------------0     MAXLRECL-------------256     EXCPEXIT----------(NULL)     CI/
CA-----------------10
       ACT-DIC-TOKEN ----X'4000010A014001C501C901D601E201E40E010EFE05FE08FE0DFE0AFE0000000000000000'
       SHROPTNS(1,3)   RECOVERY     SUBALLOC         NOERASE     COMPRESSED    CMP-ACTIVE
INDEXED       NOWRITECHK
       NOIMBED       NOREPLICAT     UNORDERED        NOREUSE     NONSPANNED
     STATISTICS
       REC-TOTAL------------721     SPLITS-CI--------------0     EXCPS-----------------54
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------6
       REC-INSERTED-----------0     FREESPACE-%CI---------15     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------7          1997.051   17:19:57
       REC-RETRIEVED-------1442     FREESPACE--------------0          X'AE40164603DFDA00'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI------245760
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI------245760
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE---------4096     HALRBA-OR-CI------245760     EXTENT-
NUMBER----------6
       DEVTYPE-------------3380     PHYRECS/TRK-----------10     HUSRBA-OR-CI------245760     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00470005'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00470005'     HI-RBA-OR-CI-------40959
       LOW-CCHH-----X'00480004'     LOW-RBA-OR-CI------40960     TRACKS-----------------1
       HIGH-CCHH----X'00480004'     HI-RBA-OR-CI-------81919
       LOW-CCHH-----X'00480005'     LOW-RBA-OR-CI------81920     TRACKS-----------------1
       HIGH-CCHH----X'00480005'     HI-RBA-OR-CI------122879
       LOW-CCHH-----X'00480006'     LOW-RBA-OR-CI-----122880     TRACKS-----------------1
       HIGH-CCHH----X'00480006'     HI-RBA-OR-CI------163839
       LOW-CCHH-----X'00480007'     LOW-RBA-OR-CI-----163840     TRACKS-----------------1
       HIGH-CCHH----X'00480007'     HI-RBA-OR-CI------204799
       LOW-CCHH-----X'00480008'     LOW-RBA-OR-CI-----204800     TRACKS-----------------1
       HIGH-CCHH----X'00480008'     HI-RBA-OR-CI------245759
   INDEX ------ TST.KSDS1.@I@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2067.295
     PROTECTION----------(NULL)
     ASSOCIATIONS
       CLUSTER--TST.KSDS1
     ATTRIBUTES
       KEYLEN-----------------10     AVGLRECL--------------0     BUFSPACE--------------0
CISIZE--------------512
       RKP--------------------0     MAXLRECL-------------505     EXCPEXIT----------(NULL)     CI/
CA-----------------46
       SHROPTNS(1,3)   RECOVERY     SUBALLOC         NOERASE     NOCOMPRESS    NOWRITECHK
NOIMBED       NOREPLICAT
       UNORDERED        NOREUSE
     STATISTICS
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE   9
```

```
                          LISTING FROM CATALOG -- TST.UCAT
       REC-TOTAL-------------7     SPLITS-CI-------------0     EXCPS-----------------58     INDEX:
       REC-DELETED-----------0     SPLITS-CA-------------0     EXTENTS---------------1
LEVELS----------------2
       REC-INSERTED----------0     FREESPACE-%CI---------0     SYSTEM-TIMESTAMP:            ENTRIES/
SECT----------3
       REC-UPDATED-----------0     FREESPACE-%CA---------0          1997.051   17:19:57     SEQ-SET-
RBA------------0
       REC-RETRIEVED---------0     FREESPACE----------19968         X'AE4016460CE87000'     HI-LEVEL-
RBA--------1024
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI-------------1     USECLASS-PRI----------0     HALRBA-OR-CI-------23552
       SPACE-SEC-------------1     USECLASS-SEC----------0     HUSRBA-OR-CI--------3584
     VOLUME
       VOLSER-----------CTS240     PHYREC-SIZE---------512     HALRBA-OR-CI-------23552     EXTENT-
NUMBER----------1
       DEVTYPE------------3380     PHYRECS/TRK----------46     HUSRBA-OR-CI--------3584     EXTENT-
TYPE--------X'00'
       VOLFLAG-----------PRIME     TRACKS/CA-------------1
       EXTENTS:
       LOW-CCHH-----X'00470006'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00470006'     HI-RBA-OR-CI-------23551
CLUSTER ------- TST.KSDS2
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.052
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       DATA-----TST.KSDS2.@D@
       INDEX----TST.KSDS2.@I@
   DATA ------- TST.KSDS2.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.052
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       CLUSTER--TST.KSDS2
     ATTRIBUTES
       KEYLEN----------------10     AVGLRECL-------------255     BUFSPACE-----------66048
CISIZE-------------32768
       RKP--------------------5     MAXLRECL-------------256     EXCPEXIT----------(NULL)     CI/
CA-----------------2
       ACT-DIC-TOKEN ----X'4000010A014001C501C901D601E201E40E010EFE05FE08FE0DFE0AFE0000000000000000'
       SHROPTNS(1,3)      SPEED     SUBALLOC         NOERASE     COMPRESSED    CMP-ACTIVE
INDEXED       NOWRITECHK
       NOIMBED       NOREPLICAT     UNORDERED        NOREUSE     NONSPANNED
     STATISTICS
       REC-TOTAL------------706     SPLITS-CI-------------0     EXCPS-----------------27
       REC-DELETED-----------0     SPLITS-CA-------------0     EXTENTS--------------10
       REC-INSERTED----------0     FREESPACE-%CI---------50     SYSTEM-TIMESTAMP:
       REC-UPDATED-----------0     FREESPACE-%CA---------50          1997.052   09:54:49
       REC-RETRIEVED--------706     FREESPACE-------------0          X'AE40F4A510898E06'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI-------------2     USECLASS-PRI----------0     HALRBA-OR-CI------655360
IDCAMS  SYSTEM SERVICES                                      TIME: 10:03:24        02/21/1997
PAGE  10
                          LISTING FROM CATALOG -- TST.UCAT
       SPACE-SEC-------------2     USECLASS-SEC----------0     HUSRBA-OR-CI------655360
     VOLUME
       VOLSER-----------CTS400     PHYREC-SIZE---------8192     HALRBA-OR-CI------655360     EXTENT-
NUMBER---------10
       DEVTYPE------------3380     PHYRECS/TRK----------5     HUSRBA-OR-CI------655360     EXTENT-
TYPE--------X'00'
       VOLFLAG-----------PRIME     TRACKS/CA-------------2
       EXTENTS:
       LOW-CCHH-----X'00010000'     LOW-RBA-OR-CI----------0     TRACKS-----------------2
       HIGH-CCHH----X'00010001'     HI-RBA-OR-CI-------65535
       LOW-CCHH-----X'0001000C'     LOW-RBA-OR-CI------65536     TRACKS-----------------2
       HIGH-CCHH----X'0001000D'     HI-RBA-OR-CI------131071
       LOW-CCHH-----X'0001000E'     LOW-RBA-OR-CI-----131072     TRACKS-----------------2
       HIGH-CCHH----X'00020000'     HI-RBA-OR-CI------196607
       LOW-CCHH-----X'00020001'     LOW-RBA-OR-CI-----196608     TRACKS-----------------2
       HIGH-CCHH----X'00020002'     HI-RBA-OR-CI------262143
       LOW-CCHH-----X'00020003'     LOW-RBA-OR-CI-----262144     TRACKS-----------------2
       HIGH-CCHH----X'00020004'     HI-RBA-OR-CI------327679
       LOW-CCHH-----X'00020005'     LOW-RBA-OR-CI-----327680     TRACKS-----------------2
       HIGH-CCHH----X'00020006'     HI-RBA-OR-CI------393215
       LOW-CCHH-----X'00020007'     LOW-RBA-OR-CI-----393216     TRACKS-----------------2
       HIGH-CCHH----X'00020008'     HI-RBA-OR-CI------458751
       LOW-CCHH-----X'00020009'     LOW-RBA-OR-CI-----458752     TRACKS-----------------2
```

```
       HIGH-CCHH----X'0002000A'     HI-RBA-OR-CI------524287
       LOW-CCHH-----X'0002000B'     LOW-RBA-OR-CI-----524288     TRACKS-----------------2
       HIGH-CCHH----X'0002000C'     HI-RBA-OR-CI------589823
       LOW-CCHH-----X'0002000D'     LOW-RBA-OR-CI-----589824     TRACKS-----------------2
       HIGH-CCHH----X'0002000E'     HI-RBA-OR-CI------655359
   INDEX ------ TST.KSDS2.@I@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.052
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       CLUSTER--TST.KSDS2
     ATTRIBUTES
       KEYLEN----------------10     AVGLRECL--------------0     BUFSPACE---------------0
CISIZE---------------512
       RKP--------------------5     MAXLRECL-------------505     EXCPEXIT----------(NULL)     CI/
CA-----------------46
       SHROPTNS(1,3)   RECOVERY     SUBALLOC          NOERASE     NOCOMPRESS    NOWRITECHK
NOIMBED       NOREPLICAT
       UNORDERED        NOREUSE
     STATISTICS
       REC-TOTAL-------------11     SPLITS-CI--------------0     EXCPS-----------------72     INDEX:
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
LEVELS-----------------2

       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:            ENTRIES/
SECT-----------1
       REC-UPDATED------------0     FREESPACE-%CA----------0          1997.052   09:54:49     SEQ-SET-
RBA------------0
       REC-RETRIEVED----------0     FREESPACE----------17920          X'AE40F4A51D877006'     HI-LEVEL-
RBA--------1024
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------23552
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE  11
                            LISTING FROM CATALOG -- TST.UCAT
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI--------5632
     VOLUME
       VOLSER------------CTS400     PHYREC-SIZE----------512     HALRBA-OR-CI-------23552     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------46     HUSRBA-OR-CI--------5632     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00010002'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00010002'     HI-RBA-OR-CI-------23551
CLUSTER ------- TST.RRDS1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.366
     PROTECTION----------(NULL)
     ASSOCIATIONS
       DATA-----TST.RRDS1.@D@
   DATA ------- TST.RRDS1.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.366
     PROTECTION----------(NULL)
     ASSOCIATIONS
       CLUSTER--TST.RRDS1
     ATTRIBUTES
       KEYLEN-----------------0     AVGLRECL-------------132     BUFSPACE------------4096
CISIZE--------------2048
       RKP--------------------0     MAXLRECL-------------132     EXCPEXIT----------(NULL)     CI/
CA-----------------90
       RECORDS/CI-------------15     MAXRECS---------31457280
       SHROPTNS(1,3)   RECOVERY     SUBALLOC          NOERASE     NOCOMPRESS      NUMBERED
NOWRITECHK       NOIMBED
       NOREPLICAT     UNORDERED     NOREUSE       NONSPANNED     NEVEREXPIR
     STATISTICS
       REC-TOTAL--------------0     SPLITS-CI--------------0     EXCPS------------------0
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------0          0000.000   00:00:00
       REC-RETRIEVED----------0     FREESPACE---------368640          X'0000000000000000'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI-------------10     USECLASS-PRI-----------0     HALRBA-OR-CI------368640
       SPACE-SEC--------------5     USECLASS-SEC-----------0     HUSRBA-OR-CI-----------0
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE---------2048     HALRBA-OR-CI------368640     EXTENT-
```

```
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------18     HUSRBA-OR-CI----------0     EXTENT-
TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------5
       EXTENTS:
       LOW-CCHH-----X'00470007'     LOW-RBA-OR-CI----------0     TRACKS----------------10
       HIGH-CCHH----X'00480001'     HI-RBA-OR-CI------368639
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE  12
                              LISTING FROM CATALOG -- TST.UCAT
CLUSTER ------- TST.UCAT
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       DATA-----VSAM.CATALOG.BASE.DATA.RECORD
       INDEX----VSAM.CATALOG.BASE.INDEX.RECORD
   DATA ------- VSAM.CATALOG.BASE.DATA.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       CLUSTER--TST.UCAT
     ATTRIBUTES
       KEYLEN-----------------44     AVGLRECL-------------505     BUFSPACE------------3072
CISIZE---------------512

       RKP--------------------0     MAXLRECL-------------505     EXCPEXIT----------(NULL)     CI/
CA-----------------92
       SHROPTNS(3,3)   RECOVERY     SUBALLOC          NOERASE     NOCOMPRESS       INDEXED
NOWRITECHK         IMBED
       NOREPLICAT     UNORDERED     NOREUSE       NONSPANNED
     STATISTICS
       REC-TOTAL-------------13     SPLITS-CI--------------0     EXCPS-----------------18
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------2
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------0          1997.051   13:50:20
       REC-RETRIEVED----------0     FREESPACE--------------0          X'AE3FE76C4FE5DC05'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------6     USECLASS-PRI-----------0     HALRBA-OR-CI-------94208
       SPACE-SEC--------------3     USECLASS-SEC-----------0     HUSRBA-OR-CI-------94208
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE----------512     HALRBA-OR-CI-------47104     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------46     HUSRBA-OR-CI-------47104     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------3
       LOW-KEY---------------00
       HIGH-KEY--------------3F
       HI-KEY-RBA----------5120
       EXTENTS:
       LOW-CCHH-----X'00000002'     LOW-RBA-OR-CI----------0     TRACKS-----------------3
       HIGH-CCHH----X'00000004'     HI-RBA-OR-CI-------47103
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE----------512     HALRBA-OR-CI-------94208     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------46     HUSRBA-OR-CI-------94208     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------3
       LOW-KEY---------------40
       HIGH-KEY--------------FF
       HI-KEY-RBA---------47104
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE  13
                              LISTING FROM CATALOG -- TST.UCAT
       EXTENTS:
       LOW-CCHH-----X'00000008'     LOW-RBA-OR-CI------47104     TRACKS-----------------3
       HIGH-CCHH----X'0000000A'     HI-RBA-OR-CI-------94207
   INDEX ------ VSAM.CATALOG.BASE.INDEX.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       CLUSTER--TST.UCAT
     ATTRIBUTES
       KEYLEN-----------------44     AVGLRECL---------------0     BUFSPACE---------------0
CISIZE-------------1024
       RKP--------------------0     MAXLRECL------------1017     EXCPEXIT----------(NULL)     CI/
```

```
CA-----------------31
       SHROPTNS(3,3)   RECOVERY     SUBALLOC         NOERASE     NOCOMPRESS    NOWRITECHK
IMBED         NOREPLICAT
       UNORDERED        NOREUSE
     STATISTICS
       REC-TOTAL-------------3     SPLITS-CI-------------0     EXCPS----------------14     INDEX:
       REC-DELETED-----------0     SPLITS-CA-------------0     EXTENTS---------------3
LEVELS----------------2
       REC-INSERTED----------0     FREESPACE-%CI---------0     SYSTEM-TIMESTAMP:           ENTRIES/
SECT----------7
       REC-UPDATED-----------0     FREESPACE-%CA---------0          1997.051   13:50:20     SEQ-SET-
RBA--------95232
       REC-RETRIEVED---------0     FREESPACE----------94208         X'AE3FE76C5BFC0405'     HI-LEVEL-
RBA----------0
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI-------------3     USECLASS-PRI----------0     HALRBA-OR-CI-------97280
       SPACE-SEC-------------3     USECLASS-SEC----------0     HUSRBA-OR-CI-------97280
     VOLUME
       VOLSER-----------CTS240     PHYREC-SIZE---------1024    HALRBA-OR-CI-------95232     EXTENT-
NUMBER----------1
       DEVTYPE------------3380     PHYRECS/TRK----------31     HUSRBA-OR-CI--------1024     EXTENT-
TYPE--------X'00'
       VOLFLAG-----------PRIME     TRACKS/CA-------------1
       EXTENTS:
       LOW-CCHH-----X'00000005'    LOW-RBA-OR-CI----------0     TRACKS----------------3
       HIGH-CCHH----X'00000007'    HI-RBA-OR-CI-------95231
     VOLUME
       VOLSER-----------CTS240     PHYREC-SIZE---------1024    HALRBA-OR-CI-------96256     EXTENT-
NUMBER----------1
       DEVTYPE------------3380     PHYRECS/TRK----------31     HUSRBA-OR-CI-------96256     EXTENT-
TYPE--------X'80'
       VOLFLAG-----------PRIME     TRACKS/CA-------------3
       LOW-KEY--------------00
       HIGH-KEY-------------3F
       EXTENTS:
       LOW-CCHH-----X'00000002'    LOW-RBA-OR-CI------95232     TRACKS----------------3

       HIGH-CCHH----X'00000004'    HI-RBA-OR-CI-------96255
     VOLUME
       VOLSER-----------CTS240     PHYREC-SIZE---------1024    HALRBA-OR-CI-------97280     EXTENT-
NUMBER----------1
       DEVTYPE------------3380     PHYRECS/TRK----------31     HUSRBA-OR-CI-------97280     EXTENT-
TYPE--------X'80'
       VOLFLAG-----------PRIME     TRACKS/CA-------------3
       LOW-KEY--------------40
       HIGH-KEY-------------FF
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE  14
                           LISTING FROM CATALOG -- TST.UCAT
       EXTENTS:
       LOW-CCHH-----X'00000008'    LOW-RBA-OR-CI------96256     TRACKS----------------3
       HIGH-CCHH----X'0000000A'    HI-RBA-OR-CI-------97279
CLUSTER ------- VSAM.COMPRESS.CONTROL
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       DATA-----VSAM.COMPRESS.CONTROL.@D@
       INDEX----VSAM.COMPRESS.CONTROL.@I@
   DATA ------- VSAM.COMPRESS.CONTROL.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
     ASSOCIATIONS
       CLUSTER--VSAM.COMPRESS.CONTROL
     ATTRIBUTES
       KEYLEN----------------44     AVGLRECL-------------128     BUFSPACE------------4608
CISIZE--------------2048
       RKP--------------------0     MAXLRECL-------------500     EXCPEXIT----------(NULL)     CI/
CA-----------------18
       SHROPTNS(4,4)   RECOVERY     SUBALLOC         NOERASE     NOCOMPRESS       INDEXED
NOWRITECHK       NOIMBED
       NOREPLICAT     UNORDERED     NOREUSE       NONSPANNED
     STATISTICS
       REC-TOTAL-------------4     SPLITS-CI-------------0     EXCPS----------------77
       REC-DELETED-----------2     SPLITS-CA-------------0     EXTENTS---------------1
       REC-INSERTED----------2     FREESPACE-%CI---------0     SYSTEM-TIMESTAMP:
       REC-UPDATED-----------5     FREESPACE-%CA---------0          1997.052   10:02:38
       REC-RETRIEVED---------50    FREESPACE-------------0          X'AE40F664633E5C01'
```

```
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI----------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------36864
       SPACE-SEC----------1     USECLASS-SEC-----------0     HUSRBA-OR-CI-------36864
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE---------2048     HALRBA-OR-CI-------36864     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------18     HUSRBA-OR-CI-------36864     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'0000000B'     HI-RBA-OR-CI-------36863
   INDEX ------ VSAM.COMPRESS.CONTROL.@I@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION----------(NULL)
IDCAMS  SYSTEM SERVICES                                          TIME: 10:03:24        02/21/1997
PAGE  15
                            LISTING FROM CATALOG -- TST.UCAT
     ASSOCIATIONS
       CLUSTER--VSAM.COMPRESS.CONTROL
     ATTRIBUTES
       KEYLEN-----------------44     AVGLRECL---------------0     BUFSPACE---------------0
CISIZE---------------512
       RKP--------------------0     MAXLRECL-------------505     EXCPEXIT----------(NULL)     CI/
CA-----------------46
       SHROPTNS(4,4)   RECOVERY     SUBALLOC         NOERASE     NOCOMPRESS    NOWRITECHK
NOIMBED       NOREPLICAT
       UNORDERED        NOREUSE
     STATISTICS
       REC-TOTAL--------------1     SPLITS-CI--------------0     EXCPS------------------65     INDEX:
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
LEVELS-----------------1
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:             ENTRIES/
SECT-----------4
       REC-UPDATED------------0     FREESPACE-%CA----------0          1997.052   10:02:38     SEQ-SET-
RBA------------0
       REC-RETRIEVED----------0     FREESPACE----------23040          X'AE40F66465AFA201'     HI-LEVEL-
RBA-----------0
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI----------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------23552

       SPACE-SEC----------1     USECLASS-SEC-----------0     HUSRBA-OR-CI---------512
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE----------512     HALRBA-OR-CI-------23552     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------46     HUSRBA-OR-CI---------512     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00470000'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00470000'     HI-RBA-OR-CI-------23551
IDCAMS  SYSTEM SERVICES                                           TIME: 10:03:24        02/21/1997
PAGE  16
                            LISTING FROM CATALOG -- TST.UCAT
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX ------------------1
                   CLUSTER ---------------8
                   DATA ------------------9
                   INDEX -----------------5
                   NONVSAM ---------------0
                   PATH ------------------1
                   SPACE -----------------2
                   USERCATALOG -----------0
                   TOTAL ----------------26
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

LISTCAT ALL Output Listing for Extra-Large Dataset

This example illustrates the LISTCAT output for an extra-large dataset, in which case CI, rather than RBA, values are maintained.

Figure 27. Example of LISTCAT ALL Output for Extra-Large Dataset

```
// EXEC
IDCAMS,SIZE=AUTO


IDCAMS  SYSTEM SERVICES                                          TIME: 13:00:46        05/15/1998
PAGE   1


 LISTCAT
-
        ALL                                 -
        ENTRIES( EXTRALARGE.KSDS.CLUSTER)  -

CATALOG(VSAM.C4G.USER.CATALOG)



CLUSTER -------
EXTRALARGE.KSDS.CLUSTER


HISTORY

       OWNER-IDENT-------(NULL)
CREATION--------1998.135
       RELEASE----------------2
EXPIRATION------1999.366
     PROTECTION----------
(NULL)
IDCAMS  SYSTEM SERVICES                                          TIME: 13:00:46        05/15/1998
PAGE   2
                          LISTING FROM CATALOG --
VSAM.C4G.USER.CATALOG
     ASSOCIATION
       DATA-----
EXTRALARGE.KSDS.CLUSTER.@D@

       INDEX----
EXTRALARGE.KSDS.CLUSTER.@I@

   DATA -------
EXTRALARGE.KSDS.CLUSTER.@D@


HISTORY

       OWNER-IDENT-------(NULL)
CREATION--------1998.135
       RELEASE----------------2
EXPIRATION------1999.366
     PROTECTION----------
(NULL)

ASSOCIATIONS

       CLUSTER--
EXTRALARGE.KSDS.CLUSTER


ATTRIBUTES

       KEYLEN-----------------8     AVGLRECL-----------32758     BUFSPACE-----------66048
CISIZE-------------32768
       RKP--------------------4     MAXLRECL----------131072     EXCPEXIT----------(NULL)     CI/
CA-----------------18
       SHROPTNS(2,3)      SPEED     SUBALLOC          NOERASE     NOCOMPRESS    EXTRALARGE
INDEXED       NOWRITECHK
       NOIMBED       NOREPLICAT     UNORDERED         NOREUSE     SPANNED
NEVEREXPIR

STATISTICS

       REC-TOTAL---------154069     SPLITS-CI--------------0
EXCPS-------------109789
       REC-DELETED------------0     SPLITS-CA--------------0
EXTENTS--------------12
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-
TIMESTAMP:
```

```
       REC-UPDATED------------0     FREESPACE-%CA----------0          1998.135
12:17:12
       REC-RETRIEVED-----154069     FREESPC-BYTES----------0
X'AEA96F45A476A001'

ALLOCATION

       SPACE-TYPE------
CYLINDER
       SPACE-PRI------------699     USECLASS-PRI-----------0     HALRBA-OR-
CI------150984
       SPACE-SEC------------699     USECLASS-SEC-----------0     HUSRBA-OR-
CI------150984

VOLUME

       VOLSER------------VSE20A     PHYREC-SIZE---------8192     HALRBA-OR-CI-------37746     EXTENT-
NUMBER----------3
       DEVTYPE-------------9345     PHYRECS/TRK------------5     HUSRBA-OR-CI------150984     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/
CA-------------15

EXTENTS:

       LOW-CCHH-----X'00010000'     LOW-RBA-OR-CI----------0
TRACKS-------------10485
       HIGH-CCHH----X'02BB000E'     HIGH-RBA-OR-
CI-----12581
       LOW-CCHH-----X'02BE0000'     LOW-RBA-OR-CI------12582
TRACKS-------------10485
       HIGH-CCHH----X'0578000E'     HIGH-RBA-OR-
CI-----25163
       LOW-CCHH-----X'057B0000'     LOW-RBA-OR-CI------25164
TRACKS-------------10485
       HIGH-CCHH----X'0835000E'     HIGH-RBA-OR-
CI-----37745

VOLUME

       VOLSER------------VSE20B     PHYREC-SIZE---------8192     HALRBA-OR-CI-------75492     EXTENT-
NUMBER----------3
       DEVTYPE-------------9345     PHYRECS/TRK------------5     HUSRBA-OR-CI------150984     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/
CA-------------15

EXTENTS:

       LOW-CCHH-----X'00010000'     LOW-RBA-OR-CI------37746
TRACKS-------------10485
       HIGH-CCHH----X'02BB000E'     HIGH-RBA-OR-
CI-----50327
       LOW-CCHH-----X'02BC0000'     LOW-RBA-OR-CI------50328
TRACKS-------------10485
       HIGH-CCHH----X'0576000E'     HIGH-RBA-OR-
CI-----62909
       LOW-CCHH-----X'05770000'     LOW-RBA-OR-CI------62910
TRACKS-------------10485
       HIGH-CCHH----X'0831000E'     HIGH-RBA-OR-
CI-----75491
IDCAMS  SYSTEM SERVICES                                          TIME: 13:00:46        05/15/1998
PAGE   3
                           LISTING FROM CATALOG -- VSAM.C4G.USER.CATALOG

VOLUME

       VOLSER------------VSE20C     PHYREC-SIZE---------8192     HALRBA-OR-CI------113238     EXTENT-
NUMBER----------3
       DEVTYPE-------------9345     PHYRECS/TRK------------5     HUSRBA-OR-CI------150984     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/
CA-------------15

EXTENTS:

       LOW-CCHH-----X'00010000'     LOW-RBA-OR-CI------75492
TRACKS-------------10485
       HIGH-CCHH----X'02BB000E'     HIGH-RBA-OR-
CI-----88073
       LOW-CCHH-----X'02BC0000'     LOW-RBA-OR-CI------88074
TRACKS-------------10485
```

```
       HIGH-CCHH----X'0576000E'     HIGH-RBA-OR-
CI----100655
       LOW-CCHH-----X'05770000'     LOW-RBA-OR-CI-----100656
TRACKS------------10485
       HIGH-CCHH----X'0831000E'     HIGH-RBA-OR-
CI----113237

VOLUME

       VOLSER------------VSE20D     PHYREC-SIZE---------8192     HALRBA-OR-CI------150984     EXTENT-
NUMBER----------3
       DEVTYPE-------------9345     PHYRECS/TRK------------5     HUSRBA-OR-CI------150984     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/
CA-------------15

EXTENTS:

       LOW-CCHH-----X'00010000'     LOW-RBA-OR-CI-----113238
TRACKS------------10485
       HIGH-CCHH----X'02BB000E'     HIGH-RBA-OR-
CI----125819
       LOW-CCHH-----X'02BC0000'     LOW-RBA-OR-CI-----125820
TRACKS------------10485
       HIGH-CCHH----X'0576000E'     HIGH-RBA-OR-
CI----138401
       LOW-CCHH-----X'05770000'     LOW-RBA-OR-CI-----138402
TRACKS------------10485
       HIGH-CCHH----X'0831000E'     HIGH-RBA-OR-
CI----150983
   INDEX ------
EXTRALARGE.KSDS.CLUSTER.@I@


HISTORY

       OWNER-IDENT-------(NULL)
CREATION--------1998.135
       RELEASE----------------2
EXPIRATION------1999.366
     PROTECTION----------
(NULL)

ASSOCIATIONS

       CLUSTER--
EXTRALARGE.KSDS.CLUSTER


ATTRIBUTES

       KEYLEN-----------------8     AVGLRECL--------------0     BUFSPACE--------------0
CISIZE---------------512
       RKP--------------------4     MAXLRECL-------------505     EXCPEXIT----------(NULL)     CI/
CA-----------------41
       SHROPTNS(2,3)   RECOVERY     SUBALLOC         NOERASE     NOCOMPRESS    NOWRITECHK
NOIMBED       NOREPLICAT
       UNORDERED        NOREUSE
NEVEREXPIR

STATISTICS

       REC-TOTAL-----------8504     SPLITS-CI-------------0     EXCPS-------------89014
INDEX:
       REC-DELETED------------0     SPLITS-CA-------------0     EXTENTS---------------12
LEVELS-----------------4
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:            ENTRIES/
SECT-----------4
       REC-UPDATED------------0     FREESPACE-%CA----------0          1998.135   12:17:12     SEQ-SET-
RBA------------0
       REC-RETRIEVED----------0     FREESPC-BYTES-----180224          X'AEA96F45E08D7C01'     HI-LEVEL-
RBA-----2561536

ALLOCATION

       SPACE-TYPE---------
TRACK
       SPACE-PRI-------------18     USECLASS-PRI-----------0     HALRBA-OR-
CI-----4534272
       SPACE-SEC-------------18     USECLASS-SEC-----------0     HUSRBA-OR-
CI-----4354048
```

```
VOLUME

       VOLSER------------VSE20A     PHYREC-SIZE----------512     HALRBA-OR-CI-----4534272     EXTENT-
NUMBER---------12
       DEVTYPE-------------9345     PHYRECS/TRK-----------41     HUSRBA-OR-CI-----4354048     EXTENT-
TYPE--------X'00'
IDCAMS  SYSTEM SERVICES                                          TIME: 13:00:46        05/15/1998
PAGE   4
                              LISTING FROM CATALOG --
VSAM.C4G.USER.CATALOG
       VOLFLAG------------PRIME     TRACKS/
CA--------------1

EXTENTS:

       LOW-CCHH-----X'02BC0000'     LOW-RBA-OR-CI----------0
TRACKS----------------18
       HIGH-CCHH----X'02BD0002'     HIGH-RBA-OR-
CI----377855
       LOW-CCHH-----X'05790000'     LOW-RBA-OR-CI-----377856
TRACKS----------------18
       HIGH-CCHH----X'057A0002'     HIGH-RBA-OR-
CI----755711
       LOW-CCHH-----X'08360000'     LOW-RBA-OR-CI-----755712
TRACKS----------------18
       HIGH-CCHH----X'08370002'     HIGH-RBA-OR-
CI---1133567
       LOW-CCHH-----X'08370003'     LOW-RBA-OR-CI----1133568
TRACKS----------------18
       HIGH-CCHH----X'08380005'     HIGH-RBA-OR-
CI---1511423
       LOW-CCHH-----X'08380006'     LOW-RBA-OR-CI----1511424
TRACKS----------------18
       HIGH-CCHH----X'08390008'     HIGH-RBA-OR-
CI---1889279
       LOW-CCHH-----X'08390009'     LOW-RBA-OR-CI----1889280
TRACKS----------------18
       HIGH-CCHH----X'083A000B'     HIGH-RBA-OR-
CI---2267135
       LOW-CCHH-----X'083A000C'     LOW-RBA-OR-CI----2267136
TRACKS----------------18
       HIGH-CCHH----X'083B000E'     HIGH-RBA-OR-CI---2644991
       LOW-CCHH-----X'083C0000'     LOW-RBA-OR-CI----2644992
TRACKS----------------18
       HIGH-CCHH----X'083D0002'     HIGH-RBA-OR-
CI---3022847
       LOW-CCHH-----X'083D0003'     LOW-RBA-OR-CI----3022848
TRACKS----------------18
       HIGH-CCHH----X'083E0005'     HIGH-RBA-OR-
CI---3400703
       LOW-CCHH-----X'083E0006'     LOW-RBA-OR-CI----3400704
TRACKS----------------18
       HIGH-CCHH----X'083F0008'     HIGH-RBA-OR-
CI---3778559
       LOW-CCHH-----X'083F0009'     LOW-RBA-OR-CI----3778560
TRACKS----------------18
       HIGH-CCHH----X'0840000B'     HIGH-RBA-OR-
CI---4156415
       LOW-CCHH-----X'0840000C'     LOW-RBA-OR-CI----4156416
TRACKS----------------18
       HIGH-CCHH----X'0841000E'     HIGH-RBA-OR-
CI---4534271

VOLUME

       VOLSER------------VSE20B     PHYREC-SIZE----------512     HALRBA-OR-CI-----------0     EXTENT-
NUMBER----------0
       DEVTYPE-------------9345     PHYRECS/TRK-----------41     HUSRBA-OR-CI-----4354048     EXTENT-
TYPE--------X'00'
       VOLFLAG--------CANDIDATE     TRACKS/
CA--------------1

VOLUME

       VOLSER------------VSE20C     PHYREC-SIZE----------512     HALRBA-OR-CI-----------0     EXTENT-
NUMBER----------0
       DEVTYPE-------------9345     PHYRECS/TRK-----------41     HUSRBA-OR-CI-----4354048     EXTENT-
TYPE--------X'00'
       VOLFLAG--------CANDIDATE     TRACKS/
CA--------------1

VOLUME
```

```
       VOLSER------------VSE20D     PHYREC-SIZE----------512     HALRBA-OR-CI----------0     EXTENT-
NUMBER----------0
       DEVTYPE-------------9345     PHYRECS/TRK----------41     HUSRBA-OR-CI-----4354048     EXTENT-
TYPE--------X'00'
       VOLFLAG--------CANDIDATE     TRACKS/
CA--------------1
IDCAMS  SYSTEM SERVICES                                          TIME: 13:00:46        05/15/1998
PAGE  5
                            LISTING FROM CATALOG --
VSAM.C4G.USER.CATALOG
         THE NUMBER OF ENTRIES PROCESSED
WAS:
                    AIX
-------------------0
                    CLUSTER
---------------1
                    DATA
-----------------1
                    INDEX
-----------------1
                    NONVSAM
---------------0
                    PATH
-----------------0
                    SPACE
-----------------0
                    USERCATALOG
-----------0
                    TOTAL
-----------------3
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS
0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS
0
IDCAMS  SYSTEM SERVICES                                          TIME: 13:00:46        05/15/1998
PAGE   6


IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS
0
1S55I  LAST RETURN CODE WAS 0000
```

LISTCAT ALLOCATION Output Listing

When you specify the LISTCAT command and include the ALLOCATION parameter, every cataloged object with space allocated to it from a VSE/VSAM data space is listed (Figure 28 on page 304). All information about the object's space is listed, but none of the object's other cataloged information is listed. The entry types that can be specified when the ALLOCATION parameter is specified are limited to DATA and INDEX.

Figure 28. Example of LISTCAT ALLOCATION Output

```
// EXEC IDCAMS,SIZE=AUTO
LIS00040

IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE   1

   LISTCAT ALLOCATION CATALOG(TST.UCAT)                                LIS00050
IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE   2
                            LISTING FROM CATALOG -- TST.UCAT
CLUSTER ------- IMP.SAM.ESDS
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2005.243
   DATA ------- TC27AB6A.VSAMDSET.DFD97051.TAE3FF4D.TC27AB6A
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2005.243
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------36864
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI--------2048
     VOLUME
       VOLSER------------CTS400     PHYREC-SIZE---------2048     HALRBA-OR-CI-------36864     EXTENT-
NUMBER----------1
```

```
       DEVTYPE-------------3380     PHYRECS/TRK-----------18     HUSRBA-OR-CI--------2048     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA-------------1
       EXTENTS:
       LOW-CCHH-----X'00010003'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00010003'     HI-RBA-OR-CI-------36863
AIX ----------- KSDS1.AIX
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
   DATA ------- KSDS1.AIX.DATA
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------40960
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI-------40960
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE---------4096     HALRBA-OR-CI-------40960     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------10     HUSRBA-OR-CI-------40960     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA-------------1
       EXTENTS:
       LOW-CCHH-----X'00480002'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00480002'     HI-RBA-OR-CI-------40959
   INDEX ------ KSDS1.AIX.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     ALLOCATION
       SPACE-TYPE---------TRACK
IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE   3
                              LISTING FROM CATALOG -- TST.UCAT
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------23552
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI---------512
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE----------512     HALRBA-OR-CI-------23552     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------46     HUSRBA-OR-CI---------512     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA-------------1
       EXTENTS:
       LOW-CCHH-----X'00480003'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00480003'     HI-RBA-OR-CI-------23551
   PATH ------- KSDS1.PATH1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
CLUSTER ------- TST.ESDS1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1997.058
   DATA ------- TST.ESDS1.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1997.058
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------4     USECLASS-PRI-----------0     HALRBA-OR-CI------163840
       SPACE-SEC--------------2     USECLASS-SEC-----------0     HUSRBA-OR-CI-------94208
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE---------4096     HALRBA-OR-CI------163840     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------10     HUSRBA-OR-CI-------94208     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA-------------2
       EXTENTS:
       LOW-CCHH-----X'00470001'     LOW-RBA-OR-CI----------0     TRACKS-----------------4
       HIGH-CCHH----X'00470004'     HI-RBA-OR-CI------163839
CLUSTER ------- TST.ESDS2
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.365
   DATA ------- TST.ESDS2.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.365
     ALLOCATION
       SPACE-TYPE---------TRACK
```

```
       SPACE-PRI--------------2     USECLASS-PRI-----------0     HALRBA-OR-CI------327680
IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE   4
                              LISTING FROM CATALOG -- TST.UCAT
       SPACE-SEC--------------2     USECLASS-SEC-----------0     HUSRBA-OR-CI------327680
     VOLUME
       VOLSER------------CTS400     PHYREC-SIZE---------4096     HALRBA-OR-CI------327680     EXTENT-
NUMBER----------4
       DEVTYPE-------------3380     PHYRECS/TRK-----------10     HUSRBA-OR-CI------327680     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------2
       EXTENTS:
       LOW-CCHH-----X'00010004'     LOW-RBA-OR-CI----------0     TRACKS-----------------2
       HIGH-CCHH----X'00010005'     HI-RBA-OR-CI-------81919
       LOW-CCHH-----X'00010006'     LOW-RBA-OR-CI------81920     TRACKS-----------------2  HIGH-CCHH----
X'00010007'
       LOW-CCHH-----X'00010008'     LOW-RBA-OR-CI-----163840     TRACKS-----------------2
       HIGH-CCHH----X'00010009'     HI-RBA-OR-CI------245759
       LOW-CCHH-----X'0001000A'     LOW-RBA-OR-CI-----245760     TRACKS-----------------2
       HIGH-CCHH----X'0001000B'     HI-RBA-OR-CI------327679
CLUSTER -------- TST.KSDS1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2067.295
   DATA -------- TST.KSDS1.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2067.295
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI------245760
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI------245760
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE---------4096     HALRBA-OR-CI------245760     EXTENT-
NUMBER----------6
       DEVTYPE-------------3380     PHYRECS/TRK-----------10     HUSRBA-OR-CI------245760     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00470005'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00470005'     HI-RBA-OR-CI-------40959
       LOW-CCHH-----X'00480004'     LOW-RBA-OR-CI------40960     TRACKS-----------------1
       HIGH-CCHH----X'00480004'     HI-RBA-OR-CI-------81919
       LOW-CCHH-----X'00480005'     LOW-RBA-OR-CI------81920     TRACKS-----------------1
       HIGH-CCHH----X'00480005'     HI-RBA-OR-CI------122879
       LOW-CCHH-----X'00480006'     LOW-RBA-OR-CI-----122880     TRACKS-----------------1
       HIGH-CCHH----X'00480006'     HI-RBA-OR-CI------163839
       LOW-CCHH-----X'00480007'     LOW-RBA-OR-CI-----163840     TRACKS-----------------1
       HIGH-CCHH----X'00480007'     HI-RBA-OR-CI------204799
       LOW-CCHH-----X'00480008'     LOW-RBA-OR-CI-----204800
TRACKS-----------------1
       HIGH-CCHH----X'00480008'     HI-RBA-OR-CI------245759
   INDEX ------ TST.KSDS1.@I@
     HISTORY
IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE   5
                              LISTING FROM CATALOG -- TST.UCAT
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------2067.295
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------23552
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI--------3584
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE----------512     HALRBA-OR-CI-------23552     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------46     HUSRBA-OR-CI--------3584     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00470006'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00470006'     HI-RBA-OR-CI-------23551
CLUSTER -------- TST.KSDS2
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.052
       RELEASE----------------2     EXPIRATION------0000.000
   DATA -------- TST.KSDS2.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.052
       RELEASE----------------2     EXPIRATION------0000.000
     ALLOCATION
       SPACE-TYPE---------TRACK
```

```
       SPACE-PRI--------------2     USECLASS-PRI-----------0     HALRBA-OR-CI------655360
       SPACE-SEC--------------2     USECLASS-SEC-----------0     HUSRBA-OR-CI------655360
     VOLUME
       VOLSER------------CTS400     PHYREC-SIZE---------8192     HALRBA-OR-CI------655360     EXTENT-
NUMBER---------10
       DEVTYPE-------------3380     PHYRECS/TRK------------5     HUSRBA-OR-CI------655360     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------2
       EXTENTS:
       LOW-CCHH-----X'00010000'     LOW-RBA-OR-CI----------0     TRACKS-----------------2
       HIGH-CCHH----X'00010001'     HI-RBA-OR-CI-------65535
       LOW-CCHH-----X'0001000C'     LOW-RBA-OR-CI------65536     TRACKS-----------------2
       HIGH-CCHH----X'0001000D'     HI-RBA-OR-CI------131071
       LOW-CCHH-----X'0001000E'     LOW-RBA-OR-CI-----131072     TRACKS-----------------2
       HIGH-CCHH----X'00020000'     HI-RBA-OR-CI------196607
       LOW-CCHH-----X'00020001'     LOW-RBA-OR-CI-----196608     TRACKS-----------------2
       HIGH-CCHH----X'00020002'     HI-RBA-OR-CI------262143
       LOW-CCHH-----X'00020003'     LOW-RBA-OR-CI-----262144     TRACKS-----------------2
       HIGH-CCHH----X'00020004'     HI-RBA-OR-CI------327679
       LOW-CCHH-----X'00020005'     LOW-RBA-OR-CI-----327680     TRACKS-----------------2
       HIGH-CCHH----X'00020006'     HI-RBA-OR-CI------393215
       LOW-CCHH-----X'00020007'     LOW-RBA-OR-CI-----393216     TRACKS-----------------2
       HIGH-CCHH----X'00020008'     HI-RBA-OR-CI------458751
       LOW-CCHH-----X'00020009'     LOW-RBA-OR-CI-----458752     TRACKS-----------------2
       HIGH-CCHH----X'0002000A'     HI-RBA-OR-CI------524287
IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE   6
                            LISTING FROM CATALOG -- TST.UCAT
       LOW-CCHH-----X'0002000B'     LOW-RBA-OR-CI-----524288     TRACKS-----------------2
       HIGH-CCHH----X'0002000C'     HI-RBA-OR-CI------589823
       LOW-CCHH-----X'0002000D'     LOW-RBA-OR-CI-----589824     TRACKS-----------------2
       HIGH-CCHH----X'0002000E'     HI-RBA-OR-CI------655359
   INDEX ------ TST.KSDS2.@I@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.052
       RELEASE----------------2     EXPIRATION------0000.000
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------23552
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI--------5632
     VOLUME
       VOLSER------------CTS400     PHYREC-SIZE----------512     HALRBA-OR-CI-------23552     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------46     HUSRBA-OR-CI--------5632     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00010002'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00010002'     HI-RBA-OR-CI-------23551
CLUSTER ------- TST.RRDS1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.366
   DATA ------- TST.RRDS1.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------1999.366
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI-------------10     USECLASS-PRI-----------0     HALRBA-OR-CI------368640
       SPACE-SEC--------------5     USECLASS-SEC-----------0     HUSRBA-OR-CI-----------0
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE---------2048     HALRBA-OR-CI------368640     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------18     HUSRBA-OR-CI-----------0     EXTENT-
TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------5
       EXTENTS:
       LOW-CCHH-----X'00470007'     LOW-RBA-OR-CI----------0     TRACKS----------------10
       HIGH-CCHH----X'00480001'     HI-RBA-OR-CI------368639
CLUSTER ------- TST.UCAT
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
   DATA ------- VSAM.CATALOG.BASE.DATA.RECORD
IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE   7
                            LISTING FROM CATALOG -- TST.UCAT
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     ALLOCATION
```

```
       SPACE-TYPE---------TRACK
       SPACE-PRI----------6     USECLASS-PRI----------0     HALRBA-OR-CI-------94208
       SPACE-SEC----------3     USECLASS-SEC----------0     HUSRBA-OR-CI-------94208
     VOLUME
       VOLSER-----------CTS240     PHYREC-SIZE----------512     HALRBA-OR-CI-------47104     EXTENT-
NUMBER----------1
       DEVTYPE----------3380     PHYRECS/TRK----------46     HUSRBA-OR-CI-------47104     EXTENT-
TYPE--------X'00'
       VOLFLAG----------PRIME     TRACKS/CA----------3
       LOW-KEY----------00
       HIGH-KEY----------3F
       HI-KEY-RBA----------5120
       EXTENTS:
       LOW-CCHH-----X'00000002'     LOW-RBA-OR-CI----------0     TRACKS----------3
       HIGH-CCHH----X'00000004'     HI-RBA-OR-CI-------47103
     VOLUME
       VOLSER-----------CTS240     PHYREC-SIZE----------512     HALRBA-OR-CI-------94208     EXTENT-
NUMBER----------1
       DEVTYPE----------3380     PHYRECS/TRK----------46     HUSRBA-OR-CI-------94208     EXTENT-
TYPE--------X'00'
       VOLFLAG----------PRIME     TRACKS/CA----------3
       LOW-KEY----------40
       HIGH-KEY----------FF
       HI-KEY-RBA---------47104
       EXTENTS:
       LOW-CCHH-----X'00000008'     LOW-RBA-OR-CI------47104     TRACKS----------3
       HIGH-CCHH----X'0000000A'     HI-RBA-OR-CI-------94207
   INDEX ------ VSAM.CATALOG.BASE.INDEX.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------2     EXPIRATION------0000.000
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI----------3     USECLASS-PRI----------0     HALRBA-OR-CI-------97280
       SPACE-SEC----------3     USECLASS-SEC----------0     HUSRBA-OR-CI-------97280
     VOLUME
       VOLSER-----------CTS240     PHYREC-SIZE---------1024     HALRBA-OR-CI-------95232     EXTENT-
NUMBER----------1
       DEVTYPE----------3380     PHYRECS/TRK----------31     HUSRBA-OR-CI--------1024     EXTENT-
TYPE--------X'00'
       VOLFLAG----------PRIME     TRACKS/CA----------1
       EXTENTS:
       LOW-CCHH-----X'00000005'     LOW-RBA-OR-CI----------0     TRACKS----------3
       HIGH-CCHH----X'00000007'     HI-RBA-OR-CI-------95231
     VOLUME
       VOLSER-----------CTS240     PHYREC-SIZE---------1024     HALRBA-OR-CI-------96256     EXTENT-
NUMBER----------1
       DEVTYPE----------3380     PHYRECS/TRK----------31     HUSRBA-OR-CI-------96256     EXTENT-
TYPE--------X'80'
       VOLFLAG----------PRIME     TRACKS/CA----------3
       LOW-KEY----------00
IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE   8
                            LISTING FROM CATALOG -- TST.UCAT
       HIGH-KEY----------3F
       EXTENTS:
       LOW-CCHH-----X'00000002'     LOW-RBA-OR-CI------95232     TRACKS----------3
       HIGH-CCHH----X'00000004'     HI-RBA-OR-CI-------96255
     VOLUME
       VOLSER-----------CTS240     PHYREC-SIZE---------1024     HALRBA-OR-CI-------97280     EXTENT-
NUMBER----------1
       DEVTYPE----------3380     PHYRECS/TRK----------31     HUSRBA-OR-CI-------97280     EXTENT-
TYPE--------X'80'
       VOLFLAG----------PRIME     TRACKS/CA----------3
       LOW-KEY----------40
       HIGH-KEY----------FF
       EXTENTS:
       LOW-CCHH-----X'00000008'     LOW-RBA-OR-CI------96256     TRACKS----------3
       HIGH-CCHH----X'0000000A'     HI-RBA-OR-CI-------97279
CLUSTER ------- VSAM.COMPRESS.CONTROL
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------2     EXPIRATION------0000.000
   DATA ------- VSAM.COMPRESS.CONTROL.@D@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------2     EXPIRATION------0000.000
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI----------1     USECLASS-PRI----------0     HALRBA-OR-CI-------36864
       SPACE-SEC----------1     USECLASS-SEC----------0     HUSRBA-OR-CI-------36864
     VOLUME
```

```
       VOLSER------------CTS240     PHYREC-SIZE---------2048     HALRBA-OR-CI-------36864     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------18     HUSRBA-OR-CI-------36864     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'0000000B'     HI-RBA-OR-CI-------36863
   INDEX ------ VSAM.COMPRESS.CONTROL.@I@
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION--------1997.051
       RELEASE----------------2     EXPIRATION------0000.000
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1     USECLASS-PRI-----------0     HALRBA-OR-CI-------23552
       SPACE-SEC--------------1     USECLASS-SEC-----------0     HUSRBA-OR-CI---------512
     VOLUME
       VOLSER------------CTS240     PHYREC-SIZE----------512     HALRBA-OR-CI-------23552     EXTENT-
NUMBER----------1
       DEVTYPE-------------3380     PHYRECS/TRK-----------46     HUSRBA-OR-CI---------512     EXTENT-
TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE   9
                            LISTING FROM CATALOG -- TST.UCAT
       LOW-CCHH-----X'00470000'     LOW-RBA-OR-CI----------0     TRACKS-----------------1
       HIGH-CCHH----X'00470000'     HI-RBA-OR-CI-------23551
IDCAMS  SYSTEM SERVICES                                          TIME: 10:47:21        02/21/1997
PAGE  10
                            LISTING FROM CATALOG -- TST.UCAT
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX ------------------1
                   CLUSTER ---------------8
                   DATA ------------------9
                   INDEX -----------------5
                   NONVSAM ---------------0
                   PATH ------------------1
                   SPACE -----------------0
                   USERCATALOG -----------0
                   TOTAL -----------------24
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

Appendix B. Format of an Alternate-Index Record

Every record in the data component of an alternate-index cluster is a variable-length logical record that contains:

1. System header information,
2. The alternate key, and
3. At least one pointer to the base cluster.

System Header Information

The system header has a fixed field length of five bytes. It contains pointer information such as the length and number of pointers.

Alternate Keys

Any field in the base cluster's records that has a fixed length and a fixed position within a record can be used as an alternate key when building an alternate index over the base cluster. If the base records span CIs, the alternate key must be in the first CI of the spanned record. If you build several alternate indexes over a base cluster, the alternate key fields of the different alternate indexes may overlap every other in the base-cluster records; they can also overlap the prime key.

In contrast to the prime key, a given alternate key may occur in several records in the base cluster. For example, if an alternate index is established by department number over a payroll file organized by employee number, all the employees with the same department number will be grouped together. This means that there will be several prime-key pointers (employee numbers) in every alternate-index record, one for every occurrence of the alternate key (department number) in the base cluster. When a given alternate key occurs in several base records, it is said to be nonunique. If it occurs in only one base record (for example, social security number), it is called unique. You must indicate whether an alternate index will contain unique or nonunique alternate keys when you define the alternate index.

Alternate-Index Pointers

The relationship between the alternate index and its base cluster is established by two different types of pointers in the alternate-index record, depending on the type of base cluster. If the base cluster is a key-sequenced file, the pointer in the alternate-index record is the prime key of the base-cluster record in which the alternate key occurs. (A prime key pointer has the same length as the prime key field of the base cluster it points to.) If the base cluster is an entry-sequenced file, the pointer is the RBA of the base-cluster record in which the alternate key occurs. (An RBA pointer is always four bytes long.) There is only one prime key or RBA pointer allowed, unless the alternate index has the NONUNIQUEKEY attribute.

For nonunique keys, pointers are associated with a given alternate key value. The pointers are ordered by their arrival time; that is, if a record in the base cluster is updated with a key change, or if a new record is inserted with the same alternate key value, VSE/VSAM adds the new prime key pointer to the end of the alternate-index record. (In the case of a key change, VSE/VSAM deletes the old pointer.)

The fact that the pointers are ordered by arrival time implies that immediately after an alternate index has been built (and as long as it remains unchanged), its pointers are in prime-key order. The maximum number of pointers that can be associated with a given alternate key is 32,767, provided the maximum possible record length for spanned records is not exceeded.

Notices

This information was developed for products and services offered in the US. This material might 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 grant 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
US

For license inquiries regarding double-byte character set (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

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 jurisdictions 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.

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 provide 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 Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

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 document 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.

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.

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 actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate 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.

Programming Interface Information

This publication documents intended Programming Interfaces that allow the customer to write programs to obtain services of z/VSE.

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 "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

IPv6/VSE is a registered trademark of Barnard Software, Inc.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

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.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Terms and Conditions for Product Documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein. IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed. You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/VSE enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using Assistive Technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/VSE. Consult the assistive technology documentation for specific information when using such products to access z/VSE interfaces.

Documentation Format

The publications for this product are in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when you use the PDF files and want to request a web-based format for a publication, you can either write an email to s390id@de.ibm.com, or use the Reader Comment Form in the back of this publication or direct your mail to the following address:

```
IBM Deutschland Research & Development GmbH
Department 3282
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany
```

In the request, be sure to include the publication number and title.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Glossary

This glossary includes terms and definitions related primarily to VSE/VSAM. If you do not find the term you are looking for, refer to the index of this book, or to the *IBM Dictionary of Computing* , SC20-1699.

The glossary includes definitions with:

- Symbol * where there is a one-to-one copy from the *IBM Dictionary of Computing*
- Symbol (A) from the *American National Dictionary for Information Processing Systems* copyright 1982 by the Computer and Business Equipment Manufacturers Association (CBEMA). Copies may be purchased from the American National Standards Institute, 1430 Broadway, New York, New York 10018. Definitions are identified by symbol (A) after definition.
- Symbols (I) or (T) from the *ISO Vocabulary - Information Processing* and the *ISO Vocabulary - Office Machines* developed by the International Organization for Standardization, Technical Committee 97, Subcommittee 1. Definitions of published sections of the vocabularies are identified by symbol (I) after the definition; definitions from draft international standards, draft proposals, and working papers in development by the ISO/TC97/SC1 vocabulary subcommittee are identified by symbol (T) after the definition, indicating final agreement has not yet been reached among participating members.

The part of speech being defined is indicated by the opening words of the descriptive phrase: "To ..." indicates a verb and "Pertaining to ..." indicates a modifier. Any other wording indicates a noun or noun phrase.

31-bit addressing
Provides addressability for address spaces of up to 2 gigabytes.

access method
A program (such as VSE/VSAM or VTAM) that allows the user to define files and addresses, and to move data to and from them. It is a technique for moving data between main storage and input/output devices.

address
1. The location in the storage of a computer where data are stored. 2. In data communication, the unique code assigned to every device or work station connected to a network.

addressing mode (AMODE)
A program attribute that refers to the address length that a program is prepared to handle on entry. Addresses may be either 24 bits or 31 bits in length. In 24-bit addressing mode, the processor treats all virtual addresses as 24-bit values; in 31-bit addressing mode, the processor treats all virtual addresses as 31-bit values.

address space
A range of up to two gigabytes of contiguous virtual storage addresses that the system creates for a user. Unlike a data space, an address space contains user data **and** programs, as well as system data and programs, some of which are common to all address spaces. Instructions execute in an address space (not a data space).

AIX
See *alternate index*.

alternate index (AIX)
In systems with VSE/VSAM, the index entries of a given base cluster organized by an alternate key, that is, a key other than the prime key of the base cluster. For example, a personnel file primarily ordered by names can be indexed also by department number.

*** alternate tape**
A tape drive to which the operating system switches automatically for tape read or write operations if the end of the volume has been reached on the originally used tape drive.

alternate track
On a direct access storage device, a track designated to contain data in place of a defective track.

AMODE
See *addressing mode*.

application program
A program written for or by a user that applies to the user's work, such as a program that does inventory control or payroll.

ASI
See *automated system initialization*.

assemble
To translate a program from assembler language into object code.

assembler
A computer program that converts assembly language instructions into object code.

*** assembler language**
A source language that includes symbolic machine language statements in which there is a on-to-one correspondence with the instruction formats and data formats of the computer.

*** automated system initialization (ASI)**
A function that allows control information for system startup to be cataloged for automatic retrieval during system startup.

*** autostart**
In z/VSE, a facility that starts-up VSE/POWER with little or no operator involvement.

*** auxiliary storage**
All addressable storage, other than main storage, that can be accessed by an input/output channel; for example, storage on magnetic tape or direct access storage devices. Synonymous with external storage and with secondary storage.

*** backup copy**
A copy of information or data that is kept in case the original is changed or destroyed.

batch processing
1. Serial processing of computer programs. 2. Pertaining to the technique of processing a set of computer programs in such a way that each is completed before the next program of the set is started. (A)

batch program
A program that is processed in series with other programs and therefore normally processes data without user interaction.

BIG-DASD
A subtype of Large DASD that has a capacity of more than 64K tracks and uses up to 10017 cylinders of the disk.

block
Usually, a block consists of several records of a file that are transmitted as a unit. But if records are very large, a block can also be part of a record only. On an FBA disk, a block is a string of 512 bytes of data. In the (E)CKD environment, blocks are called records. See also *control block*.

buffer
An area of storage temporarily reserved for input or output operations; an area into which data is read or from which data is written. Synonymous with I/O area.

byte
Eight adjacent binary digits that are operated upon as a unit and that constitute the smallest addressable unit of information within a computer system. Normally, it represents a stored character.

catalog
A directory of files and libraries, with reference to their locations. A catalog may contain other information such as the types of devices in which the files are stored, passwords, and blocking factors. (I) (A)

See also *VSE/VSAMmaster catalog*.

channel program
One or more channel command words that control a sequence of data channel operations. Execution of this sequence is initiated by a single start I/O (SIO) instruction.

CA
See *control area*.

CI
See *control interval*.

CKD
See *count-key-data (CKD) device*.

close
The function that ends the connection between a file and a program, and ends the processing. Contrast with open.

cluster
In systems with VSE/VSAM, a named structure consisting of a group of related components; for example, a data component with its index component.

component
1. Hardware or software that is part of a computer system. 2. A functional part of an operating system, for example: job control program, VSE/POWER. 3. In VSE/VSAM, a named, cataloged group of stored records, such as the data component or index component of a key-sequenced file or alternate index.

conditional job control
The capability of the job control program to process or to skip one or more statements based on a condition that is tested by the program.

configuration
The devices and programs that make up a system, subsystem, or network.

control area (CA)
In VSE/VSAM, a group of control intervals (CIs) used as a unit for formatting a data set before adding records to it. Also, in a key-sequenced data set, the set of control intervals pointed to by a sequence-set index record (which, for example, is used by VSE/VSAM for distributing free space).

control block
An area within a program or a routine defined for the purpose of storing and maintaining control information.

control interval (CI)
A fixed length area of disk storage where VSE/VSAM stores records and creates distributes free space. It is the unit of information that VSE/VSAM transfers to or from disk storage.

count-key-data (CKD) device
A disk storage device that stores data in the record format: count field, key field, data field. The count field contains, among others, the address of the record in the format: cylinder, head (track), record number and the length of the data field. The key field, if present, contains the record's key (search argument). CKD disk space is allocated by tracks and cylinders. Contrast with fixed-block-architecture (FBA) device.

DASD
See *direct access storage device*.

Data Facility Product (DFP)
See *DFSMSdfp*.

data import
The process of reformatting data that was used under one operating system (for example, IBM System/3) such that it can subsequently be used under a different operating system (for example, the IBM z/VSE system).

data management
A major function of the operating system. It involves organizing, storing, locating, and retrieving data.

data set
1. The major unit of data storage and retrieval, consisting of a collection of data in one or several prescribed arrangements and described by control information to which the system has access. See also *file*.

*** default value**
A value assumed when no value has been specified. Synonymous with assumed value.

device
A hardware component of a computer system with a specific purpose.

device address
1. The identification of an input/output device by its channel and unit number. 2. In data communication, the identification of any device to which data can be sent or from which data can be received.

*** device class**
The generic name for a group of device types, for example, all display stations belong to the same device class. Contrast with device type.

*** device type**
The name of a particular kind of device; for example, 9345 (IBM DASD module), 3420 (IBM magnetic tape unit). Contrast with device class.

DFP
See *DFSMSdfp*.

DFSMSdfp
DFSMSdfp (formerly known as DFP) provides data management, device support, program library management, utility functions, and support for the z/OS operating system.

direct access
Accessing data on a storage device using their address and not their sequence. This is the typical access on disk devices as opposed to magnetic tapes. Contrast with sequential access.

*** direct access storage**
A storage device that provides direct access to data. (1)

*** direct access storage device (DASD)**
A device in which access time is effectively independent of the location of the data. See also *fixed-block-architecture (FBA) device*.

directory
1. A table of identifiers and references to the corresponding items of data. (I) (A) 2. In z/VSE, an index that is used by the system to locate one or more sequential blocks of program information that are stored on direct access storage.

disk
Synonymous for magnetic disk.

diskette
A thin, flexible magnetic disk and a semi-rigid protective jacket, in which the disk is permanently enclosed.

disk sharing
An option that lets independently operating computer systems to jointly use common data residing on shared disk devices.

dump
1. To record, at a particular instant, the contents of all or part of one storage device in another storage device. Dumping is usually for the purpose of debugging. (T) 2. Data that has been dumped. (T) 3. To copy data in a readable format from main or auxiliary storage onto an external medium such as tape, diskette, or printer. 4. To copy the contents of all or part of virtual storage for the purpose of collecting error information.

Enterprise Storage Server (ESS)
An IBM disk storage system designed for performance, automation, integration, and continuous data availability. It supports advanced copy functions, which can be critical for increasing data availability by providing important disaster recovery and backup protection.

*** entry-sequenced file**
A VSE/VSAM file whose records are loaded without respect to their contents and whose relative byte addresses cannot change. Records are retrieved and stored by addressed access, and new records are added to the end of the file.

entry-sequenced data set (ESDS)
An entry-sequenced file under VSE/VSAM. Its records are loaded without respect to their contents, and whose relative byte addresses cannot change. Records are retrieved and stored by addressed access, and new records are added to the end of the file. See also *SAM ESDS file*.

exit
1. To execute an instruction within a portion of a computer program in order to terminate the execution of that portion. 2. See *user exit routine*.

exit routine
1. Either of two types of routines: installation exit routines or user exit routines. Synonymous with exit program. 2. See *user exit routine*.

extra-large dataset
A KSDS which can exceed 4 GB in size.

extent
Continuous space on a disk or diskette that is occupied by or reserved for a particular file or VSE/VSAM data space.

*** external storage**
Storage that is accessible to a processor only through input-output channels. An external storage may sometimes be considered as peripheral equipment. Synonymous with auxiliary storage. (T)

FAT-DASD
A subtype of Large DASD, it supports a device with more than 4369 cylinders (64K tracks) up to 64K cylinders.

FBA
See *fixed-block-architecture (FBA) device*.

FBA block
A unit of data that is transported as a unit on FBA disk devices.

Fibre Channel Protocol (FCP)
A combination of hardware and software conforming to the Fibre Channel standards and allowing system and peripheral connections via FICON® and FICON Express® feature cards on IBM zSeries processors. In z/VSE, zSeries FCP is employed to access industry-standard SCSI disk devices.

file
A named set of records stored or processed as a unit. (T) Synonymous with data set.

fixed-block-architecture (FBA) device
A disk storage device that stores data in blocks of fixed size. These blocks are addressed by block number relative to the beginning of the particular file. Contrast with count-key-data (CKD) device.

generation
See *macrogeneration*.

generic name
The initial characters common to names, that can be used to identify a group of items. A generic name usually ends with an *; for example, ORD* identifies all items whose names begin with the characters ORD.

index
1. A table used to locate records in an indexed sequential data set or an indexed file. 2. In VSE/VSAM, an ordered collection of pairs, each consisting of a key and a pointer, used by VSE/VSAM to sequence

and locate the records of a key-sequenced data set or file; it is organized in levels of index records. See also *alternate index*.

initial program load (IPL)
The process of loading system programs and preparing the system to run jobs.

input
1. Pertaining to a functional unit or channel involved in an input process, or to the data involved in such process. 2. Loosely, input data, input process. 3. Information or data to be processed.

input/output (I/O)
See *input* and *output*.

IPL
See *initial program load*.

JCL
See *job control language*.

job
One program, or a group of related programs called job steps, complete with the JCL statements necessary for a particular run. A job is identified in the job stream by a JOB statement followed by one EXEC statement for each of the programs or job steps.

job catalog
A catalog made available for a job by using the file name IJSYSUC in the respective DLBL statement.

job control language (JCL)
A language that serves to prepare a job or each job step of a job to be run. Some of its functions are: to identify the job, to determine the I/O devices to be used, set switches for program use, log (or print) its own statements, and fetch the first phase of each job step.

job control statement
A particular statement of JCL.

job step
One of a group of related programs complete with the JCL statements necessary for a particular run. Every job step is identified in the job stream by an EXEC statement under one JOB statement for the entire job.

job stream
The sequence of jobs as submitted to an operating system.

*** KB (kilobyte)**
1024 bytes of storage.

key
In VSE/VSAM, one or several characters taken from a certain field (key field) in data records for identification and sequence of index entries or of the records themselves.

key sequence
The collating sequence of either records themselves or of their keys in the index, or the collating sequence of records and keys. The key-sequence is alphanumeric.

key-sequenced data set (KSDS)
Under VSE/VSAM, a key-sequenced file. See *key-sequenced file*.

key-sequenced file
A VSE/VSAM file whose records are loaded in key sequence and controlled by an index. Records are retrieved and stored by keyed access or by addressed access, and new records are inserted in the file in key sequence.

KSDS
See *key-sequenced data set*.

*** label**
A record that identifies a volume on tape, disk, or diskette, or that identifies a file on the volume.

Large DASD
A DASD device with a capacity exceeding 65535 (64K) tracks.

local shared resources (LSR)
1. An option for sharing I/O buffers, I/O-related control blocks, and channel programs among VSE/VSAM data sets in a resource pool that serves one partition or address space. 2. A VSE/VSAM option activated by the macros BLDVRP, DLVRP, and WRTBFR to share control blocks among files.

logical record
A user record, normally pertaining to a single subject and processed by data management as a unit. Contrast with physical record which may be larger or smaller.

LSR
See *local shared resources*.

macro
See *macroinstruction*

macrodefinition
A set of statements and instructions that defines the name of, format of, and conditions for generating a sequence of assembler statements and machine instructions from a single source statement.

macroexpansion
See *macrogeneration*.

macrogeneration
An operation in which an assembler produces a sequence of assembler language statements by processing a macrodefinition called by a macroinstruction. Macrogeneration takes place before assembly. Synonymous with macroexpansion.

macroinstruction
1. In assembler programming, an assembler language statement that causes the assembler to process a predefined set of statements called a macrodefinition. The statements normally produced from the macrodefinition replace the macroinstruction in the program.

magnetic tape
A tape with a magnetizable layer on which data can be stored. (T)

*** MB (megabyte)**
One megabyte equals 1,048,576 bytes.

maximum (max) CA
A unit of allocation equivalent to the maximum control area size on a count-key-data or fixed-block device. On a CKD device, the max CA is equal to one cylinder.

message
1. In z/VSE, a communication sent from a program to the operator or user. It can appear on a console, a display terminal, or in a printout.

*** migrate**
To move to a changed operating environment, usually to a new release or version of a system.

minimum (min) CA
A unit of allocation equivalent to the minimum control area size on a count-key-data or fixed-block device. On a CKD device, the min CA is equal to one track.

module
A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading; for example, the input to, or output from an assembler, compiler, linkage editor, or executive routine. (A)

online processing
Processing by which the input data enters the computer directly from a display station and the output data is transmitted directly to the display station.

open
To connect a file or a library to a program for processing. Contrast with close.

*** operating system**
Software that controls the execution of programs and that may provide services such as resource allocation, scheduling, input/output control, and data management. Although operating systems are predominantly software, partial hardware implementations are possible. (T)

*** operator command**
A statement to a control program, issued via a console or terminal. It causes the control program to provide requested information, alter normal operations, initiate new operations, or end existing operations.

output
1. Pertaining to a functional unit or channel involved in an output process, or to the data involved in such process. 2. Loosely, output data, output process. 3. Information or data that has been processed.

*** partition**
In z/VSE, a division of the virtual address area that is available for program execution.

*** password**
In computer security, a string of characters known to the computer system and a user. The user must specify it to gain full or limited access to the system and to the data stored in it.

path
1. In ACF/VTAM, the intervening nodes and data links connecting a terminal and an application program in the host processor. 2. In VSE/VSAM, a named logical entity providing access to the records of a base cluster either directly or through an alternate index.

*** physical record**
The amount of data transferred to or from auxiliary storage. Synonymous with block.

processor
The hardware component that interprets and executes instructions.

*** processor storage**
1. The storage provided by one or more processing units. 2. In virtual storage systems, synonymous with real storage.

*** random processing**
1. The treatment of data without respect to its location in external storage, and in an arbitrary sequence governed by the input against which it is to be processed. 2. The processing of records in an order other than the order in which they exist in a file.

*** read**
To acquire or interpret data from a storage device, from a data medium, or from another source. (I) (A)

*** real storage**
The main storage in a virtual system. Physically, real storage and main storage are identical. Conceptually however, real storage represents only part of the range of addresses available to the user of a virtual storage system. Traditionally, the total range of addresses available to the user was provided by the main storage. (I) (A)

record
A collection of related data or words, treated as a unit. See *logical record* and *physical record*.

*** recover**
After an execution failure, to establish a previous or new status from which execution can be resumed. (T)

relative-record data set (RRDS)
A VSE/VSAM file whose records are loaded into fixed-length slots and accessed by the relative-record numbers of these slots.

residency mode (RMODE)
A program attribute that refers to the location where a program is expected to reside in virtual storage.

restore
To write back on disk data that was previously written from disk to an intermediate storage medium such as tape.

RMODE
See *residency mode*.

routine
A program, or part of a program, that may have some general or frequent use. (T)

*** RPG II**
A commercially oriented programming language suitable specifically designed for writing application programs intended for business data processing.

RRDS
See *relative-record data set*.

run
1. A performance of one or more jobs or programs. (I) (A) 2. To cause a program, utility, or other machine function to be performed.

SAM
See *sequential access method*.

SAM ESDS file
A SAM file managed in VSE/VSAM space, so it can be accessed by both SAM and VSE/VSAM macros.

SCSI
A peripheral interface originally introduced for small computers. IBM Enterprise Storage Server (ESS) disks can be accessed via a SCSI interface implemented via zSeries Fibre Channel Protocol (FCP) channels.

SDL
See *system directory list*.

sequential access
The serial retrieval of records in their entry sequence or serial storage of records with or without a premeditated order. Contrast with direct access.

sequential access method (SAM)
A data access method that writes to and reads from an I/O device record after record (or block after block). On request, the support performs device control operations such as line spacing or page ejects on a printer, or skip a certain number of tape marks on a tape drive.

sequential file
A file in which records are processed in the order in which they are entered and stored.

service program
A program in general support of computer processes, for example, a diagnostic program, a trace program, or a sort program. (T)

*** shared virtual area (SVA)**
In z/VSE, a high address area of virtual storage that contains a system directory list (SDL) of frequently used phases, resident programs that can be shared between partitions, and an area for system support.

*** spanned record**
A logical record contained in more than one block.

split
To double a specified unit of storage space (CI or CA) dynamically when the specified minimum of free space gets used up by new records.

*** standard label**
A fixed-format record that identifies a volume of data such as a tape reel or a file that is part of a volume of data.

startup
The process of performing IPL of the operating system and of getting all subsystems and application programs ready for operation.

storage
A device, or part of a device, that can retain data. See also *auxiliary storage, processor storage, virtual storage*.

*** suballocated file**
A VSE/VSAM file that occupies a portion of an already defined data space. The data space may contain other files. Contrast with unique file.

SVA
See *shared virtual area*.

system directory list (SDL)
A list containing directory entries of frequently-used phases and of all phases resident in the SVA. The list resides in the SVA.

track
A circular path on the surface of a disk or diskette. Smallest unit of physical disk space.

*** unique file**
A VSE/VSAM file that occupies a data space of its own. The data space is defined at the same time as the file and cannot contain any other file. Contrast with suballocated file.

unit of transfer
The amount of data that can be transferred between virtual storage and an I/O device in response to a read or write request.

user exit routine
A user-written routine that receives control at predefined user exit points.

*** utility program**
1. A computer program in general support of computer processes; for example, a diagnostic program, a trace program, or a sort program. (T) Synonymous with service program. 2. A program that performs an everyday task such as copying data from one storage device to another.

variable-length relative-record data set (VRDS)
A VSE/VSAM relative-record data set with variable-length records. See also *relative-record data set*.

*** virtual address**
The address of a location in virtual storage. A virtual address must be translated into a real address in order to process the data in processor storage.

*** virtual address area**
The area available as a program address range.

virtual address space
In z/VSE, a subdivision of the virtual address area available to the user for the allocation of private, non-shared partitions.

virtual disk
A range of up to two gigabytes of contiguous virtual storage addresses that a program can use as workspace. Although the virtual disk exists in storage, it appears as a real FBA disk device to the user program. All I/O operations directed to a virtual disk are intercepted and the data to be written to, or read from, the disk is moved to or from a data space.

Like a data space, a virtual disk can hold only user data; it does not contain shared areas, system data or programs. Unlike an address space or a data space, data is not directly addressable on a virtual disk. To manipulate data on a virtual disk, the program has to perform I/O operations.

virtual storage
Addressable space image for the user from which instructions and data are mapped into processor storage locations.

virtual storage access method (VSAM)
An access method for indexed or sequential processing of fixed and variable length records on direct access devices. The records in a VSE/VSAM data set or file can be organized: in logical sequence using a key field (key sequence); in physical sequence in which they are written on the data set or file (entry sequence); by using relative record numbers.

These and other functions are provided by the IBM product VSE/VSAM.

volume
A data carrier mounted and demounted as a unit; for example, a reel of magnetic tape, a disk pack. (I) Some disk units have no demountable packs. In that case, a volume is the portion available to one read/write mechanism.

volume identifier
The volume serial number, which is a number in a volume label assigned when a volume is prepared for use by the system. See *volume serial number*.

*** volume serial number**
A number in a volume label assigned when a volume is prepared for use in a system.

volume table of contents (VTOC)
A table on a disk volume that describes every file on it.

VRDS
See *variable-length relative-record data set*.

VSE/VSAM
Virtual Storage Extended/Virtual Storage Access Method. See *virtual storage access method*.

VSAM
See *virtual storage access method*.

*** VSE/VSAM managed space**
A user-defined space on disk that is under the control of VSE/VSAM.

*** VSE/VSAM master catalog**
A key-sequenced data set or file with an index containing extensive data set and volume information that VSE/VSAM requires to locate data sets or files, allocate and deallocate storage space, verify the authorization of a program or operator to gain access to a data set or file, and to accumulate use statistics for data sets or files.

*** VSE/VSAM user catalog**
An optional VSE/VSAM catalog used in the same way as the master catalog and pointed to by the master catalog. Use of user catalogs lessens the contention for the master catalog and facilitates volume portability.

VSE
Virtual Storage Extended. 1. Synonym for VSE/Advanced Functions. 2. An operating system that is an extension of Disk Operating System/Virtual Storage (DOS/VSE). The current designation is z/VSE.

VSE/Advanced Functions
The basic operating support needed for a VSE-controlled installation. Synonymous with VSE.

VSE/ESA
The predecessor of z/VSE.

VTOC
See *volume table of contents*.

work file
A file used to for temporary storage of data being processed.

z/OS
The primary operating system for IBM eServer zSeries processors.

z/VSE
The current designation of the VSE operating system, formerly referred to as VSE/ESA.

Index

A

access control 156, 161, 171, 183
access control - logging and reporting (ACLR) 12
accessibility 317
ACLR (VSE/Access Control - Logging and Reporting) 12
adding records through the REPRO command 31
adding volumes 66
ADDVOLUMES parameter
    in ALTER command 66
alias 15
ALL parameter
    in LISTCAT command 169
allocating space
    by range of key values 25
allocation
    of free space 21
ALLOCATION parameter
    in LISTCAT command 169
ALTER command
    allocation parameters 65
    data organization parameters 65
    entry types that can be altered 60
    examples 38, 222
    format 60
    name parameter 65
    parameters 66
    protection and integrity parameters 65
    specifying information that alters an entry 38
altering catalog entries 38, 60
alternate index (AIX)
    back up 50
    building an 34
    creating 33
    defining an
        examples of 33
    path, defining 32
    record, building 33
    record, format of 311
    restoration 50
    support (BACKUP/RESTORE) 49
    upgrading 36
ALTERNATEINDEX parameter
    in DEFINE command 83
    in DELETE command 150
    in LISTCAT command 169
AN print chain option 205
AREAS parameter
    in PARM command 205
assign
    tapes (input/output)
    12
assigning data space to a catalog 16
association between data (LISTCAT output) 272
ATTEMPTS parameter
    in ALTER command 66
    in DEFINE
ATTEMPTS parameter *(continued)*
    in DEFINE *(continued)*
        ALTERNATEINDEX command 83
        CLUSTER command 102
        MASTERCATALOG command 123
        USERCATALOG command 142
attributes
    altering 38
    changing 38
    defining 15
    nullifying 66
attributes used, listing on 273
AUTHORIZATION parameter
    in ALTER command 66
    in DEFINE
        ALTERNATEINDEX command 83
        CLUSTER command 102
        MASTERCATALOG command 123
        PATH command 132
        USERCATALOG command 142
    nullifying 66

B

BACKIN, file name for RESTORE 55
backing up objects
    BACKUP command 72
    batch environment 50
    buffer allocation 244
    buffer size/allocation 73
    catalog (copy information) 51
    deletion of old copy 54
    error handling 52
    example job streams 244
    excluding objects 50
    file to disk 53
    file to tape 53
    generic backup 242
    generic backup, excluding objects from 243
    generic restore, excluding objects from 253
    how to use functions (examples) 240
    listings 240
    multiprogramming environment 50
    prerequisite 59
    requirements for files 49
    several objects 50
    start/stop mode 50
    streaming mode 50
    streaming mode, specifying 54
    support for 49
    to disk 50
    to tape 50
    using generic names 50, 73, 242
BACKOUT, file name for BACKUP 53
BACKUP command
    // DLBL statements 54
    alternate indexes, back up of 50

BACKUP command *(continued)*
alternate tape support 240
backing up several objects 50
backing up to disk 50
backing up to tape 50
BLOCKSIZE parameter 244
buffer allocation 244
buffer size 244
buffer size/allocation 73, 194
BUFFERS parameter 244
buffers, number of 244
catalog migration (restriction) 51
CATALOG parameter 73
control areas, reorganizing 50
copy catalog information 51
description 72
error handling 52
example job streams 244
EXCLUDE parameter 50
file assignment (JCL) 53
file modification 191
file name (JCL) 53, 55
file type (JCL) 53, 55
FlashCopy on ESS 252
format 72
generic backup 242
generic backup, excluding objects from 243
generic backup, specifying 242
generic names, using 50, 73
global/local specification 191
job control 53, 54
keywords, use of 191
labels supported 53
listings 240
multiple catalogs 251
NOREWIND parameter 73
overview 48
parameters 73
paths, back up of 50
performance considerations 191
prerequisite 59
requirements 49
SOURCEVOLUMES parameter 73
space (primary), allocation 194
space (secondary), allocation 194
space allocation for data component 194
space class, modifying 194
STDLABEL parameter and TLBL 53
SYNONYMCATALOG parameter 73
SYNONYMLIST parameter 73
TARGETVOLUMES parameter 73
unique files 191
using 53, 240
when to use 53
BACKUP/RESTORE and EXPORT/IMPORT, when to use 53
Backup/Restore Function
// DLBL statements (BACKUP) 54
// DLBL statements (RESTORE) 55
alternate index 49
alternate tape support 240
BACKUP command (description) 72
backup functions 49
BACKUP/RESTORE, when to use 53
buffer allocation (BACKUP) 244
Backup/Restore Function *(continued)*
buffer allocation (RESTORE) 257
buffer size/allocation 73
catalog backup/restore 51
catalog migration (restriction) 51
control areas, reorganizing 50
copy, deletion of old 54
cross-reference listings 240
deletion of old copy 54
devices supported 50
empty object 49
error handling 52
example job streams (BACKUP) 244
example job streams (RESTORE) 258
explained 48
EXPORT/IMPORT, when to use 53
file space allocation, changing 51
functions (BACKUP), examples of using 240
functions (overview) 49
functions (RESTORE), examples of using 253
generic BACKUP 242
generic names, using 50
generic RESTORE 253
global/local specification (RESTORE) 191
job control statements required (BACKUP) 54
job control statements required (RESTORE) 55
KSDS, reorganization of data component 50
listings 240
objects supported 49
performance considerations 52, 191
RESTORE command (description) 189
restore functions 49
SAM ESDS files 49
what you can do 49
base cluster 15
basic information (IDCAMS commands)
coding conventions 4
explanation to presentation 4
notational conventions 4
syntax 6
types 3
basic information (IDCAMS)
call from other program 1
coding conventions for commands 4
commands, types of 3
functions and commands 1
introduction 1
invoke, how to 1
tasks you can do 1
what you can do 1
basic information on VSE/VSAM
tape, process/create files on 12
batch environment (BACKUP) 50
beginning location
in PRINT command 171
in REPRO command 183
bigger catalog 41
BLDINDEX command
examples 34, 223
format 77
parameters 77
blocks on a volume, listing on 283
BLOCKS parameter in DEFINE

BLOCKS parameter in DEFINE *(continued)*
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
SPACE command 136
USERCATALOG command 142
BLOCKSIZE subparameter
in EXPORT command 156
in IMPORT command 161
in PRINT command 171
in REPRO command 183
buffer space, I/O
access method services, used by 11
buffers (BACKUP), fixing in real storage 54
BUFFERSPACE parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
building an alternate index 34

C

call IDCAMS from other program 1
CANCEL command 79
CANDIDATE parameter
in DEFINE SPACE command 136
candidate volume 23, 66
catalog
assigning data space to a 16
backing up 40
copy information (BACKUP/RESTORE) 51
DEDICATE parameter for a 17
defining a 16
defining a master 16
defining a user 16
defining objects in a 15
master 16
ORIGIN parameter for a 18
reloading a 41
reorganizing, how to 40
space estimates 16
unloading a 40
use in data and space management 15
user 15
catalog check service aid (IKQVCHK) 150
catalog entries
altering 38
defining 15
deleting 39
listing 47
types that can be altered 60
catalog entries flagged 'not usable' 150
CATALOG parameter
in ALTER command 66
in BACKUP command 73
in BLDINDEX command 77
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
NONVSAM command 130
PATH command 132
SPACE command 136
CATALOG parameter *(continued)*
in DEFINE *(continued)*
USERCATALOG command 142
in DELETE command 150
in IMPORT command 161
in LISTCAT command 169
catalog space estimates 16
catalog use
in data and space management 15
catalog, enlarging 41
cataloging
alternate indexes 33
clusters 22
data spaces 19
entry-sequenced files 29
files 22
key-sequenced files 25
nonVSAM files 37
paths 35
relative-record files 30
CHAIN parameter
in PARM command 205
changing attributes 38
CHARACTER parameter
in PRINT command 171
CIMODE parameter
in EXPORT command 156
CLASS parameter in DEFINE
MASTERCATALOG command 123
SPACE command 136
USERCATALOG command 142
clauses, null 203
CLUSTER parameter
in DEFINE command 102
in DELETE command 150
in LISTCAT command 169
CODE parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
PATH command 132
USERCATALOG command 142
nullifying 66
coding of IDCAMS commands 4
command execution, controlling 201
command format
functional 59
modal 201
command symbols 4
commands
ALTER 60
BACKUP 72
BLDINDEX 77
CANCEL 79
DEFINE
ALTERNATEINDEX 79
CLUSTER 97
MASTERCATALOG 120
NONVSAM 129
PATH 131
SPACE 135
USERCATALOG 139

commands *(continued)*
DELETE 148
DO 204
END 204
EXPORT 155
functional 59
IF 201
IMPORT 160
LISTCAT 167
modal 201
PARM 205
PRINT 171
REPRO 181
RESTORE 189
SNAP 199
terminator 9
VERIFY 200
commands (IDCAMS)
creating files on tape 12
ENVIRONMENT parameter (tape processing) 12
files on tape, processing/creating 12
multifile volume considerations 13
multivolume file, how to specify 13
opening a file for access by PRINT or REPRO 11
procedure, creating output files on a volume 13
procedure, creating/processing files on unlabeled tape 14
procedure, processing multifile volume 13
processing files on tape 12
tape processing 12
condition codes (IF-THEN-ELSE) 201
condition codes, setting 204
CONNECT parameter
in IMPORT command 161
connecting a user catalog to the master catalog 44, 161
control areas, reorganizing 50
CONTROLINTERVALSIZE parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
controlling command execution 201
CONTROLPW parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
PATH command 132
USERCATALOG command 142
nullifying 66
conventions (IDCAMS commands)
coding commands 4
comments, specifying 7
continuation errors, common 8
continuation of statement, specifying 8
default columns for entering verbs 7
default columns, resetting 7
explanation to presentation 4
keyword parameters 7
parameter, meaning 6
parameters set 7
parameters, specifying 7
password, specifying 7
positional parameters 7
conventions (IDCAMS commands) *(continued)*
structure 6
subparameters 7
subparameters set 7
syntax 6
terminator, meaning 6
verb entry, default columns for 7
verb, meaning 6
verb, specifying 7
conventions, command 4
conversion of files 181
COPY parameter
in SNAP command 199
copying
catalogs 40, 181
examples 232
files 181
COUNT parameter
in PRINT command 171
in REPRO command 183
creating
alternate indexes 33
backup copies of catalogs 40
catalogs 16
clusters 22
data spaces 19
entry-sequenced files 29
files 22
key-sequenced files 25
master catalogs 15
nonVSAM files 37
paths 35
relative-record files 30
suballocated files 23
unique files 24
user catalogs 15
cross-partition sharing 83, 102
cross-system sharing 83, 102
cylinders on a volume, listing on 283
CYLINDERS parameter in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
SPACE command 136
USERCATALOG command 142

D

DA, file type for RESTORE 55
DASD
support for Large DASD 14
data
portability 46
records, printing 47
data compression
attributes 273
CMP suffix 242
COMPRESSION attribute in DEFINE command 102
copy information (BACKUP/RESTORE) 51
example of a deleted catalog with objects 39
excluding objects in backup/restore 50
exporting a cluster - RECORDMODE 156
FORCE attribute of DELETE 150
IMPORT command 160

data compression *(continued)*
OBJECT TYPE 242
restrictions 102
DATA parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
USERCATALOG command 142
in LISTCAT command 169
data processing statistics, listing on 279
data sets stored on volumes, listing on 276
data space
defining 19
examples 20
for catalog 16
data space used on volumes, listing on 276
data spaces on a volume, listing on 283
DATARECORDS parameter 51
DATASET parameter
in VERIFY command 200
DATAVOLUMES parameter 255
DDSR parameter
in SNAP command 199
debugging tool (PARM command) 205
DEDICATE parameter
in DEFINE
MASTERCATALOG command 123
SPACE command 136
USERCATALOG command 142
using the 17
default catalog
CATALOG (IMPORT) 161
CATALOG (LISTCAT) 169
CATALOG parameter 66, 83
CATALOG parameter (DELETE) 150
CATALOG parameter (NONVSAM) 130
CATALOG parameter (PATH) 132
DATASET (VERIFY) 200
entryname (EXPORT) 156
INDATASET parameter 77
MODEL parameter 83, 142
MODEL parameter (PATH) 132
MODEL parameter (SPACE) 136
OUTDATASET parameter 77
where explained 59
WORKVOLUME parameter 77
default margins 205
DEFAULTVOLUMES parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
in IMPORT command 161
DEFINE command
ALTERNATEINDEX
examples 33, 223
format 79
parameters 83
CLUSTER
examples 23, 212, 215
format 97
parameters 102
MASTERCATALOG
examples 17, 209
DEFINE command *(continued)*
MASTERCATALOG *(continued)*
format 120
parameters 123
NONVSAM
examples 37, 215
format 129
parameters 130
PATH
examples 36, 223
format 131
parameters 132
SPACE
examples 20, 210
format 135
parameters 136
USERCATALOG
examples 17, 210
format 139
parameters 141
defining
alternate indexes 33, 79
catalogs 16, 120, 139
clusters 22, 97
data spaces 19, 135
files 22, 97
nonVSAM files 37, 129
objects in a catalog 15
paths 35, 131
spaces 19, 135
VRDS 30
VSE/VSAM data space 19
VSE/VSAM file (cluster) 22
work files on virtual disk 37
DELETE command
examples 39, 233, 237
format for deleting a catalog 148
format for deleting a cluster or alternate index 148
format for deleting a NonVSAM object 148
format for deleting a path 148
format for deleting a space 148
parameters 150
deleting
catalog entries 39, 148
catalogs 148
device dependencies 14
DEVICETYPE subparameter
in IMPORT command 161
DEVICETYPES parameter
in DEFINE NONVSAM command 130
diagnosis tools (PARM command) 205
disability 317
DISCONNECT parameter
in EXPORT command 156
distributed free space 21
DLBL statement (job control)
opening a file for access by PRINT or REPRO 11
DO-END command sequence 204
DUMP character
in PRINT command 171
DUMP parameter
in PRINT command 171
dump points 205
dumps 205

dynamic file 22

E

ELSE clause 201
END, in DO-END command sequence 204
ending location
  in PRINT command 171
  in REPRO command 183
Enterprise Storage Server (ESS) 3, 59, 72
ENTRIES parameter
  in LISTCAT command 169
entry-sequenced file, defining an 29
entryname/password parameter
  in ALTER command 66
  in DELETE command 150
  in EXPORT command 156
ENVIRONMENT parameter (tape processing) 12
ENVIRONMENT subparameter
  in EXPORT command 156
  in IMPORT command 161
  in PRINT command 171
  in REPRO command 183
EQ (equal) comparand 201
ERASE parameter
  in ALTER command 66
  in DEFINE
    ALTERNATEINDEX command 83
    CLUSTER command 102
  in DELETE command 150
  in EXPORT command 156
  in IMPORT command 161
erasing a file 39
error handling (VSE/VSAM Backup/Restore Function) 52
errors in continuation specification (IDCAMS commands) 8
examples
  ALTER command 38, 222
  BACKUP functions, using 240
  BLDINDEX command 34, 223
  continuation specification errors 8
  cross-reference listings (BACKUP/RESTORE) 240
  DEDICATE parameter 17
  DEFINE
    ALTERNATEINDEX command 33, 223
    CLUSTER command 23, 212, 215, 225
    MASTERCATALOG command 17, 209
    NONVSAM command 37, 215
    PATH command 36, 223
    SPACE command 20, 210, 225
    USERCATALOG command 17, 210
  DELETE command 39, 233, 237
  EXPORT command 227, 228
  IMPORT command 229, 230
  job streams (BACKUP) 244
  job streams (RESTORE) 258
  LISTCAT ALL output listing 289, 299
  LISTCAT ALLOCATION output listing 304
  LISTCAT command 212, 215, 222
  LISTCAT output listing (no parameters specified) 284
  LISTCAT output listings 284
  LISTCAT SPACE ALL output listing 288
  LISTCAT VOLUME output listing 285
  ORIGIN parameter 18
  PRINT command 218, 221, 239
examples *(continued)*
  REPRO command 31, 218, 221, 232
  RESTORE functions, using 253
  work files on virtual disk 37
EXCEPTIONEXIT parameter
  in ALTER command 66
  in DEFINE
    ALTERNATEINDEX command 83
    CLUSTER command 102
    MASTERCATALOG command 123, 136
    USERCATALOG command 142
execution, controlling command 201
EXPORT command
  format 155
  parameters 156
  use of 44, 46
EXPORT/IMPORT and BACKUP/RESTORE, when to use 53
EXPORT/IMPORT, used for transporting or backing up files 44
extents, listing on 281
EXTERNALSORT parameter
  in BLDINDEX command 77
Extra-large dataset
  LISTCAT example 299
extracting catalog information 46
EXTRALARGEDATASET parameter 102

F

Fibre Channel Protocol (FCP) 14
file
  accessibility, verifying 48
  backing up 44
  backing up, devices supported 50
  backing up, requirements 49
  cataloging a 22
  copying a 40
  creating a 22
  defining a 22
  deleting a 39
  dynamic 22
  entry-sequenced 29
  erasing a 39
  key-sequenced 25
  loading records into a 31
  nonVSAM 37
  on virtual disk 37
  portable 46
  relative-record 30
  reorganizing a 42
  space allocation (RESTORE), changing 51
  spanned 102
  specifying information that defines a 23
  suballocated 22, 23
  transporting a 44
  types of 22
  unallocated 22
  unique 24
FILE parameter
  in DEFINE
    ALTERNATEINDEX command 83
    CLUSTER command 102
FILE subparameter
  in IMPORT command 161

files
commands for creating files on tape 12
commands for processing files on tape 12
creating on tape, commands for 12
more than one on a volume 13
multivolume, how to specify for 13
processing on tape, commands for 12
tape, commands for creating on 12
tape, commands for processing from 12
files stored on volumes, listing on 276
FILESEQUENCENUMBERS parameter
in DEFINE NONVSAM command 130
FOR parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
PATH command 132
USERCATALOG command 142
FORCE parameter
in DELETE command 150
free space, distributed 21
free space, listing on 279
FREESPACE parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
FROMADDRESS parameter
in PRINT command 171
in REPRO command 183
FROMKEY parameter
in PRINT command 171
in REPRO command 183
FROMNUMBER parameter
in PRINT command 171
in REPRO command 183
functional commands (IDCAMS) 3, 59
functions of IDCAMS 1, 59

G

GE (greater or equal) comparand 201
generic back up (BACKUP) 242
generic names 242
generic names using with BACKUP/RESTORE 50
generic restore (RESTORE) 253
GRAPHICS parameter
in PARM command 205
GT (greater than) comparand 201

H

HEX parameter
in PRINT command 171
high allocated RBA or CI 272
high used RBA or CI 272
HINDEXDEVICE subparameter
in PRINT command 171
in REPRO command 183
hints on specifying free space 21
history on objects 278
HN print chain option 205
how to read railroad track diagrams 4

I

IDCAMS Interactive Interface (IDCONS) 55
IDCAMS utility program
call from other program 1
coding conventions for commands 4
commands and functions (overview) 1
commands, types of 3
functional commands 3
functions and commands (overview) 1
introduction 1
invoke, how to 1
modal commands 4
tape, considerations for files on 12
IDCONS (IDCAMS Interactive Interface) 55
IF command
continuation of a statement, coding 8
continuation specification errors 8
IF statements, nested 203
IF-THEN-ELSE command sequence
examples 203, 209, 212, 215, 218, 221, 222
format of 201
IGNOREERROR parameter
in DELETE command 150
IKQVCHK (catalog check service aid) 150
IMPORT command
examples 229, 230
format 160
parameters 161
use of 44, 46
INDATASET parameter
in BLDINDEX command 77
index CI size, changing 254
INDEX parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
USERCATALOG command 142
in LISTCAT command 169
index processing statistics, listing on 279
INDEXED parameter
in DEFINE CLUSTER command 102
INDEXFILE parameter 194
INFILE parameter
in IMPORT command 161
in PRINT command 171
in REPRO command 183
INHIBIT parameter
in ALTER command 66
INHIBITSOURCE parameter
in EXPORT command 156
INHIBITTARGET parameter
in EXPORT command 156
input files on tape, processing of 13
INTERNALSORT parameter
in BLDINDEX command 77
interpreting LISTCAT output 271
invoke IDCAMS, how to 1
ISAM to VSE/VSAM conversion 181

J

job control statements
  backing up file to disk 53
  backing up file to tape 53
  backup file assignment 53
  buffers (BACKUP), fixing in real storage 54
  creating backup file 54
  examples (BACKUP) 244
  examples (RESTORE) 258
  file name (JCL for BACKUP) 53, 55
  file type (JCL for BACKUP) 53, 55
  labels supported (BACKUP) 53
  opening a file for access by IDCAMS 11
  restoring backup file 55
  UPSI statement (BACKUP) 54
job streams, samples 209

K

key-sequenced file, defining a 25
KEYRANGES parameter
  in DEFINE
    ALTERNATEINDEX command 83
    CLUSTER command 102
KEYRANGES subparameter
  in IMPORT command 161
KEYS parameter
  in ALTER command 66
  in DEFINE
    ALTERNATEINDEX command 83
    CLUSTER command 102
keyword parameters (IDCAMS command), specifying 7

L

label
  not supported 12
labels supported (BACKUP) 53
Large DASD 14
LASTCC parameter
  in IF-THEN-ELSE clause 201
  in SET command 204
LE (less or equal) comparand 201
list of parameters (IDCAMS command), specifying a 7
list of subparameters (IDCAMS command), specifying a 7
LISTCAT command
  examples 212, 215, 222
  format 167
  messages on output 284
  output listings, examples of 284
  output listings, interpreting 271
  parameters 169
LISTCAT output listings
  allocated space 272
  allocation group 272
  association group 272
  attributes group 273
  attributes used 273
  blocks on a volume 283
  code with field groups 271
  cylinders on a volume 283
LISTCAT output listings *(continued)*
  data processing statistics 279
  data space group (volume) 276
  data space used on volumes 276
  examples 284
  extents, information on 281
  files stored on volumes 276
  free space information 279
  groups of information 271
  high allocated RBA or CI 272
  high used RBA or CI 272
  history group 278
  index processing statistics 279
  interpreting 271
  LISTCAT ALL specified 289, 299
  LISTCAT ALLOCATION specified 304
  LISTCAT SPACE ALL specified 288
  LISTCAT VOLUME specified 285
  messages 284
  no parameters specified 284
  nonVSAM entry 278
  nonVSAM file, where stored 278
  object creation/expiration dates 278
  object owner, identifier of 278
  objects protected 279
  protection group 279
  relationship of entries 272
  space on a volume 283
  statistics group 279
  tracks on a volume 283
  volume content 281
  volume entry special fields 283
  volume group 281
  volume space (data component) 281
  volume space (index component) 281
listing catalog entries 47
load and reload, catalog 40
loading a portable file 46
loading records into a file 31
loading VSE/VSAM files 31
logical unit assignments for tape 12
LT (less than) comparand 201

M

making a file portable 46
MARGINS parameter
  in PARM command 205
margins, standard 205
master catalog
  creating a 15, 16
  examples 17
MASTERCATALOG parameter
  in DEFINE command 120
  in DELETE command 150
MASTERPW parameter
  in ALTER command 66
  in DEFINE
    ALTERNATEINDEX command 83
    CLUSTER command 102
    PATH command 132
    USERCATALOG command 142
  nullifying 66
MAXCC parameter

MAXCC parameter *(continued)*
in IF-THEN-ELSE clause 201
in SET command 204
merging VSE/VSAM files 181
messages with LISTCAT output 284
migrating
files, commands for 12
modal commands (IDCAMS) 4, 201
MODEL parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
PATH command 132
USERCATALOG command 142
modeling 15
modifying sequence of execution 201
MODULE subparameter
in ALTER command 66
moving user catalogs and files between systems 40
multifile tape volume
creating, consideration for 13
processing, consideration for 13
multifile volume
considerations 13
procedure for creating 13
procedure for processing 13
procedure, creating/processing files on unlabeled tape 14
specify, how to 13
multifile volume considerations for IDCAMS 13
multifile volumes 12, 13, 156, 228
multiprogramming environment (BACKUP) 50

N

NAME parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
NONVSAM command 129
PATH command 131
USERCATALOG command 139
naming
alternate indexes 83
clusters 102
data components 102
index components 102
master catalogs 123
nonVSAM files 129
paths 131
user catalogs 139
NE (not equal) comparand 201
nested IF statements 203
NEWNAME parameter
in ALTER command 66
NEWNAME subparameter
in IMPORT command 161
NOALLOCATION parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
NOCIFORMAT SAM ESDS file 102
NOCOPY parameter
in SNAP command 199
nonVSAM files
cataloging 129
copying 181
defining 37
deleting 150
examples 37, 215
listing catalog entries for 169
printing 171
where stored, information on 278
NONVSAM parameter
in DEFINE command 129
in DELETE command 150
in LISTCAT command 169
NOPROMPT parameter
in SNAP command 199
NOREWIND parameter
in BACKUP command 73
in RESTORE command 194
notations, command 4
notusable entries flagged 150
NOTUSABLE parameter
in LISTCAT command 169
null clause with modal commands (IDCAMS) 8
null clause, inadvertent specification 8
null clauses 203
NULLIFY parameter
in ALTER command 66
NUMBERED parameter
in DEFINE CLUSTER command 102

O

object (tasks, commands)
backup copy, deletion of 54
buffer allocation (BACKUP) 244
buffer allocation (RESTORE) 257
catalog backup/restore 51
catalog, copy information 51
cross-reference listings (BACKUP/RESTORE) 240
example job streams (BACKUP) 244
example job streams (RESTORE) 258
exclude (BACKUP/RESTORE) 50
job control 11
supported (BACKUP/RESTORE) 49
using generic names (BACKUP) 242
using generic names (RESTORE) 253
objects (history of), listing on 278
OBJECTS parameter
in IMPORT command 161
operation
creating files on tape, commands for 12
ENVIRONMENT parameter (tape processing) 12
processing files on tape, commands for 12
tape assignment 12
tape, process/create files on 12
unlabeled tape processing with ACLR 12
ORDERED parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
ORDERED subparameter
in IMPORT command 161

ORIGIN parameter
in DEFINE
MASTERCATALOG command 123
SPACE command 136
USERCATALOG command 142
using the 18
OUTDATASET parameter
in BLDINDEX command 77
OUTFILE parameter
in EXPORT command 156
in REPRO command 183
output files on tape, creating 13
output from PRINT DUMP 177
OUTPW parameter
in IMPORT command 161
OWNER parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
PATH command 132
USERCATALOG command 142

P

parameters set (IDCAMS command), specifying a 7
PARM command
default columns, resetting 7
partition dump 205
password requirements 9
passwords
specify, how to 7
PATH parameter
in DEFINE command 131
in DELETE command 148
in LISTCAT command 167
path, defining a 35
PATHENTRY parameter
in DEFINE PATH command 132
permanent exportation 44
PN print chain option 205
portable files 46
positional parameters (IDCAMS command), specifying 7
prime index 32
PRIMEDATADEVICE subparameter
in EXPORT command 156
in IMPORT command 161
in PRINT command 171
in REPRO command 183
print chain options 205
PRINT command
examples 218, 221, 239
format 171
parameters 171
sample output from 177
PRINT DUMP output 177
printing data records 47
procedures
create/process files on unlabeled tape 14
creating output files on a volume 13
process multifile volume 13
PROMPT parameter
in SNAP command 199
protection of objects, listing on 279
protection parameters
in ALTER command 65
in DEFINE
ALTERNATEINDEX command 82
CLUSTER command 101
MASTERCATALOG command 122
PATH command 131
PURGE parameter
in DELETE command 150
in EXPORT command 156
in IMPORT command 161

Q

QN print chain option 205

R

railroad track diagrams 4
RBA (relative byte address)
allocated, information on 272
loading records into a file 31
printing records by RBA 47
used, information on 272
with alternate index 32
reading LISTCAT output 271
READPW parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
PATH command 132
USERCATALOG command 142
nullifying 66
RECMAP command 177
record format (alternate-index) 311
record loading (into a file) 31
RECORDFORMAT subparameter
in PRINT command 171
in REPRO command 183
records
printing of 47
RECORDS parameter in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
SPACE command 136
USERCATALOG command 142
RECORDSIZE parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
SPACE command 136
in PRINT command 171
in REPRO command 183
RECOVERY parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
RECOVERY parameter of DEFINE command 72

RELATE parameter
in DEFINE ALTERNATEINDEX command 83
relative-record file
defining a 30
reloading a catalog 41
REMOVEVOLUMES parameter
in ALTER command 66
removing volumes 66
renaming files 66, 161
reorganizing a file 42
reorganizing catalogs 40
REPLACE parameter
in REPRO command 31, 183
REPRO command
adding record 31
description 181
examples 31, 218, 221, 232
for catalog backup and reorganizing a file 40
format 181
loading a file 31
overview 3
parameters 183
reorganizing a file 42
reorganizing files 21
resetting condition codes 204
respecifying attributes 60
RESTORE command
// DLBL statements 55
alternate indexes, restoration 50
automatic definition for restoration 54
buffer allocation 257
BUFFERS parameter 257
buffers, number of 257
catalog migration (restriction) 51
control areas, reorganizing 50
copy catalog information 51
DATARECORDS parameter 51
deletion of old copy 54
description 189
error handling 52
example job streams 258
EXCLUDE parameter 50
file space allocation, changing 51
format 189
generic name 194
generic names, using 50
generic restore 194, 253
generic restore, excluding objects from 253
job control 54, 55
migrating object to another device 50
multiple catalogs 269
NOREWIND parameter 194
overview 48
parameters (details) 194
parameters (summary) 191
paths, restoration 50
prerequisite 59
restoring several objects 50
target volume 50
using 54
using functions 253
when to use 53
restoring objects
automatic definition for restoration 54
restoring objects *(continued)*
buffer allocation 257
catalog (copy information) 51
deletion of old copy 54
error handling 52
example job streams 258
exclude 50
file modification (RESTORE) 191
files, changing space allocation 51
generic name 194
how to use functions (examples) 253
prerequisite 59
RESTORE command 189
several objects 50
support for 49
to other device 50
to other volume 50
using generic names 50, 194, 253
retention date for files
nullifying 66
RETENTION parameter
in ALTER command 66
reusable files 102
REUSE parameter
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
in REPRO command 183
REWIND subparameter
in EXPORT command 156
in IMPORT command 161
in PRINT command 171
in REPRO command 183
RN print chain option 205

S

SAM ESDS files (BACKUP/RESTORE) 49
SAM files and data compression 102
SAM to VSE/VSAM conversion 181
sample job streams 209
sample output from PRINT DUMP 177
SCRATCH parameter
in DELETE command 150
SCSI disk devices 14
SD, file type for BACKUP 53
sequence of execution, controlling 204
SET command 204
set of parameters (IDCAMS command), specifying a 7
set of subparameters (IDCAMS command), specifying a 7
setting condition codes 204
SHAREOPTIONS parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
sharing data
options modified in ALTER command 66
specified in DEFINE command 83, 102
SKIP parameter
in PRINT command 171
in REPRO command 183
SN print chain option 205
SNAP command

SNAP command *(continued)*
format 199
parameters 199
SOURCEVOLUMES parameter
in BACKUP command 73
in SNAP command 199
space allocated, listing on 272
space allocation to VSE/VSAM objects 20
space estimates for catalog 16
space for data component on volume, listing on 281
space for index component on volume, listing on 281
space on a volume, listing on 283
SPACE parameter
in DEFINE command 136
in DELETE command 150
in LISTCAT command 169
space, data
allocating 135
defining 19, 135
for catalogs 16
work files in 37
SPANNED parameter
in DEFINE CLUSTER command 102
start/stop mode (BACKUP) 50
starting location
in PRINT command 171
in REPRO command 183
STDLABEL subparameter
in EXPORT command 156
in IMPORT command 161
in PRINT command 171
in REPRO command 183
STEP parameter
in CANCEL command 79
stopping location
in PRINT command 171
in REPRO command 183
storage allocation, changing 254, 268
storage requirements for VSE/VSAM catalog 16
streaming mode (BACKUP) 50, 54
STRING subparameter
in ALTER command 66
structure of IDCAMS commands 6
suballocated file, defining a 22, 23
subparameters set (IDCAMS command), specifying a 7
subparameters to keywords (IDCAMS command), specifying 7
SYNONYMCATALOG parameter
in BACKUP command 73
SYNONYMLIST parameter
in BACKUP command 73
syntax
diagrams, understanding 4
of commands 4
of IDCAMS commands 6
symbols 4
syntax checking (PARM command) 205
SYS004 161, 171, 183
SYS005 156, 183
SYS005 (backup file assignment) 53

T

TABLE parameter
in PARM command 205
TARGETVOLUMES parameter
in BACKUP command 73
in SNAP command 199
tasks you can do with IDCAMS commands 1
temporary exportation 44
TEMPORARY parameter
in EXPORT command 156
terms explained 319
THEN clause 201
TN print chain option 205
TOADDRESS parameter
in PRINT command 171
in REPRO command 183
TOKEY parameter
in PRINT command 171
in REPRO command 183
TONUMBER parameter
in PRINT command 171
in REPRO command 183
TRACE parameter
in PARM command 205
trace tables 205
tracing output 205
tracks on a volume, listing on 283
TRACKS parameter in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
SPACE command 136
USERCATALOG command 142
transporting files between systems 44
types of files 22

U

unallocated file 22
UNINHIBIT parameter
in ALTER command 66
unique file
defining a 22, 24
space for a 20
UNIQUE parameter in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
UNIQUEKEY parameter
in ALTER command 66
in DEFINE ALTERNATEINDEX command 83
unlabeled tape, creating output file 14
unlabeled tape, processing input file 14
UNLOAD subparameter
in EXPORT command 156
in IMPORT command 161
in PRINT command 171
in REPRO command 183
unloading a catalog 40
UPDATE parameter
in ALTER command 66
UPDATEPW parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102

UPDATEPW parameter *(continued)*
in DEFINE *(continued)*
MASTERCATALOG command 123
PATH command 132
USERCATALOG command 142
nullifying 66
UPGRADE parameter
in ALTER command 66
in DEFINE ALTERNATEINDEX command 83
UPSI statement 244
UPSI statement (BACKUP), use of 54
USECLASS parameter
in DEFINE AIX command 83
in DEFINE CLUSTER command 102
in IMPORT command 161
user catalog
backing up 40
connecting to master catalog 161
creating a 16
defining a
examples 17, 210
disconnecting from master catalog 156
on virtual disk 142
portability 40
reloading a 40
unloading a 40
user security verification routine (USVR)
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
PATH command the entry point of the USVR 132
USERCATALOG command 142
USERCATALOG parameter
in DEFINE command 142
in DELETE command 150
in LISTCAT command 169

V

variable-length relative record data set (VRDS) 101
variable-length relative-record data set (VRDS), defining a 30
VERIFY command
format 200
parameters 200
verifying a file's accessibility 48
virtual disk support (example) 37
virtual disk support (restrictions) 83, 102, 123, 136, 142
virtual tape 102
volume content, listing on 281
VOLUME parameter
in DEFINE
MASTERCATALOG command 123
USERCATALOG command 142
in LISTCAT command 169
VOLUMES parameter 255
VOLUMES parameter in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
NONVSAM command 130
SPACE command 136
VOLUMES subparameter
in IMPORT command 161
VRDS (variable-length relative record data set) 101
VRDS, defining a 30
VRDS, relationship to RRDS and KSDS 30
VSE/Access Control - Logging and Reporting 156, 161, 171, 183
VSE/Access Control - Logging and Reporting (ACLR) 12
VSE/ACLR 156, 161, 171, 183
VSE/VSAM processing, statistics on 279
VSE/VSAM to SAM conversion 181

W

words explained 319
work files on virtual disk (example) 37
worksheet, for catalog space estimates 16
WORKVOLUMES parameter
in BLDINDEX command 77
WRITECHECK parameter
in ALTER command 66
in DEFINE
ALTERNATEINDEX command 83
CLUSTER command 102
MASTERCATALOG command 123
USERCATALOG command 142
WRITECHECK parameter of DEFINE command 72

Product Number: 5686-VS6

SC34-2707-01