IBM IMS Library Integrity Utilities for z/OS
Version 2 Release 2

User's Guide

IBM
Note:
Before using this information and the product it supports, read the "Notices" topic at the end of this information.

Fourth Edition (September 2017)
This edition applies to Version 2 Release 2 of IBM IMS Library Integrity Utilities for z/OS (program number 5655-U08) and to all subsequent releases and modifications until otherwise indicated in new editions.
This edition replaces SC19-3979-02.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
Contents

About this information .................................. vii

Chapter 1. IMS Library Integrity Utilities overview .............................................. 1
What’s new in IMS Library Integrity Utilities .................................................. 2
IMS Library Integrity Utilities terminology ..................................................... 6
What does IMS Library Integrity Utilities do? .............................................. 8
IMS Library Integrity Utilities solutions ......................................................... 13
Functional enhancements in IMS Library Integrity Utilities Version 2 ............ 15
  Functional enhancements in Version 2 Release 2 ........................................ 15
  Functional enhancements in Version 2 Release 1 ........................................ 17
Service updates and support information ................................................. 20
Product documentation and updates ......................................................... 21
Accessibility features for IMS Library Integrity Utilities ..................................... 23

Chapter 2. Configuring IMS Library Integrity Utilities ............................................. 25
Hardware and software prerequisites ......................................................... 26
Configuring for initial installation ............................................................... 27
  Setting up security for Consistency Checker and Multiple Resource Checker ... 27
Migration procedures .................................................................................. 28
  Migrating Integrity Checker .................................................................... 28
  Migrating Advanced ACBGEN .................................................................. 33

Chapter 3. Integrity Checker utility ................................................................. 35
Integrity Checker overview ............................................................................. 36
Planning for Integrity Checker configuration .............................................. 39
  LIU module load library customization ...................................................... 39
  LICON data sets and global option modules ............................................. 40
  Integrity Checker configuration requirements ....................................... 41
  Runtime options and environments ......................................................... 47
  Historical data maintained in LICON data sets ........................................ 48
  Considerations for activating Integrity Checker ....................................... 50
Activating Integrity Checker .......................................................................... 54
  Setting up the global option modules ....................................................... 54
  Setting up the LICON data sets ............................................................... 55
  Setting up RACF security ....................................................................... 58
  Customizing LIU load modules ............................................................... 59
Configuring for a BPE-based DBRC environment ......................................... 65
Verifying that Integrity Checker is activated ............................................... 65
Restarting IMS online and running IMS batch application, IMS utility, and IMS Tools jobs ................................................................. 66
Maintaining Integrity Checker ......................................................................... 67
  Maintaining RDEs ..................................................................................... 67
  Maintaining global option modules ......................................................... 75
  Maintaining LICON data sets ................................................................. 77
  Restarting Integrity Checker after an abend .............................................. 77
  Applying PTFs to IMS Library Integrity Utilities and to IMS .................... 78
Preventing database corruption with Integrity Checker ..................................... 79
Restrictions: Cases where DMB verification is not done .................................. 79
DMB mismatch in IMS online environment or application jobs ....................... 80
DMB mismatch during database maintenance and operation ......................... 83
Addressing a DMB mismatch ...................................................................... 84
Deactivating Integrity Checker ...................................................................... 86
Deactivating Integrity Checker when IMS Library Integrity Checker is installed as a stand-alone product ................................................................. 86
Deactivating Integrity Checker when IMS Library Integrity Utilities is installed as a component of an IMS tools solution pack .................................................. 87
Output from Integrity Checker ...................................................................... 89
  FABLPRT data set .................................................................................... 89
  FABLSNAP data set ................................................................................ 90
Global option module generation macro .................................................... 91
  Creating global option modules ............................................................ 91
JCL requirements for the FABLPGEN program ............................................. 92
LICON utility reference ............................................................................... 98
  JCL requirements for the LICON utility ................................................. 98
  Input for the LICON utility ................................................................... 100
  INIT.DB command ............................................................................... 101
  INIT.LICON command ......................................................................... 106
  CHANGE.DB command ....................................................................... 106
  DELETE.DB command .......................................................................... 110
  EXPIRE.DB command ......................................................................... 112
  LIST.DB command .............................................................................. 113
  LIST.LICON command ........................................................................ 114
  RECOVER.DB command ....................................................................... 115
  VERIFY.DB command .......................................................................... 116
  Output from the LICON utility ............................................................... 118

Chapter 4. Consistency Checker utility .............................................................. 125
Consistency Checker utility overview .......................................................... 126
Restriction for Consistency Checker ............................................................ 129
Checking the consistency of definitions ....................................................... 130
JCL requirements for the Consistency Checker utility .................................... 131
  Control statements for the Consistency Checker utility .......................... 131
JCL examples for the Consistency Checker utility ........................................ 134
  Example: Checking the consistency of DBDs ........................................... 137
  Example: Checking the consistency of PSBs .......................................... 138
  Example: Checking the consistency of DBDs and PSBs .......................... 139
Output from the Consistency Checker utility .............................................. 140
  SYSOUT data set ............................................................................... 140
  SYSPRINT data set ............................................................................ 140

Chapter 5. Multiple Resource Checker utility ................................................... 157
Multiple Resource Checker utility overview .............................................. 158
<table>
<thead>
<tr>
<th>Chapter 6. DBD/PSB/ACB Compare utility</th>
<th>199</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD/PSB/ACB Compare utility overview</td>
<td>200</td>
</tr>
<tr>
<td>Restrictions and considerations for the DBD/PSB/ACB Compare utility</td>
<td>202</td>
</tr>
<tr>
<td>Comparing IMS control blocks</td>
<td>203</td>
</tr>
<tr>
<td>JCL requirements for the DBD/PSB/ACB Compare utility</td>
<td>204</td>
</tr>
<tr>
<td>Control statements for the DBD/PSB/ACB Compare utility</td>
<td>207</td>
</tr>
<tr>
<td>DBD, PSB, ACB control statements</td>
<td>209</td>
</tr>
<tr>
<td>REPORT control statement</td>
<td>212</td>
</tr>
<tr>
<td>NOCOMP control statement</td>
<td>213</td>
</tr>
<tr>
<td>JCL examples for the DBD/PSB/ACB Compare utility</td>
<td>223</td>
</tr>
<tr>
<td>Example: Comparing two DBDs</td>
<td>223</td>
</tr>
<tr>
<td>Example: Comparing two DBDs that have different names</td>
<td>223</td>
</tr>
<tr>
<td>Example: Comparing a DBD with a DBD-type ACB</td>
<td>224</td>
</tr>
<tr>
<td>Example: Comparing two PSBs</td>
<td>224</td>
</tr>
<tr>
<td>Example: Comparing a PSB with a PSB-type ACB</td>
<td>225</td>
</tr>
<tr>
<td>Example: Comparing two ACBs</td>
<td>225</td>
</tr>
<tr>
<td>Example: Comparing ACBs with DBDs and PSBs</td>
<td>226</td>
</tr>
<tr>
<td>Example: Comparing DBDs, PSBs, and ACBs</td>
<td>226</td>
</tr>
<tr>
<td>Output from the DBD/PSB/ACB Compare utility</td>
<td>227</td>
</tr>
<tr>
<td>SYSOUT data set</td>
<td>227</td>
</tr>
<tr>
<td>SYSPRINT data set</td>
<td>227</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 7. DBD/PSB/ACB Mapper utility</th>
<th>243</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD/PSB/ACB Mapper utility overview</td>
<td>244</td>
</tr>
<tr>
<td>Restrictions for the DBD/PSB/ACB Mapper utility</td>
<td>246</td>
</tr>
<tr>
<td>Printing hierarchical structure of databases</td>
<td>247</td>
</tr>
<tr>
<td>JCL requirements for the DBD/PSB/ACB Mapper utility</td>
<td>248</td>
</tr>
<tr>
<td>Control statements for the DBD/PSB/ACB Mapper utility</td>
<td>250</td>
</tr>
<tr>
<td>JCL examples for the DBD/PSB/ACB Mapper utility</td>
<td>253</td>
</tr>
<tr>
<td>Example: Generating DBD maps</td>
<td>253</td>
</tr>
<tr>
<td>Example: Generating PSB maps</td>
<td>254</td>
</tr>
<tr>
<td>Example: Generating ACB maps</td>
<td>255</td>
</tr>
<tr>
<td>Example: Generating DBD, PSB, ACB maps</td>
<td>256</td>
</tr>
<tr>
<td>Example: Creating a DBD and generating a DBD map</td>
<td>257</td>
</tr>
<tr>
<td>Example: Creating a PSB and generating a PSB map</td>
<td>258</td>
</tr>
<tr>
<td>Output from the DBD/PSB/ACB Mapper utility</td>
<td>259</td>
</tr>
<tr>
<td>SYSOUT data set</td>
<td>259</td>
</tr>
<tr>
<td>SYSPRINT data set</td>
<td>259</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 8. DBD/PSB/ACB Reversal utility</th>
<th>279</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD/PSB/ACB Reversal utility overview</td>
<td>280</td>
</tr>
<tr>
<td>Restrictions for the DBD/PSB/ACB Reversal utility</td>
<td>283</td>
</tr>
<tr>
<td>Converting IMS control blocks to control statements</td>
<td>284</td>
</tr>
<tr>
<td>JCL requirements for the DBD/PSB/ACB Reversal utility</td>
<td>285</td>
</tr>
<tr>
<td>Control statements for the DBD/PSB/ACB Reversal utility</td>
<td>288</td>
</tr>
<tr>
<td>JCL examples for the DBD/PSB/ACB Reversal utility</td>
<td>300</td>
</tr>
<tr>
<td>Example: Re-creating the sources from DBDs and PSBs</td>
<td>300</td>
</tr>
<tr>
<td>Example: Re-creating the sources from ACBs</td>
<td>301</td>
</tr>
<tr>
<td>Example: Obtaining DBD library information</td>
<td>302</td>
</tr>
<tr>
<td>Example: Obtaining PSB library information</td>
<td>302</td>
</tr>
<tr>
<td>Example: Obtaining control statement source and Mapper input</td>
<td>303</td>
</tr>
<tr>
<td>Output from the DBD/PSB/ACB Reversal utility</td>
<td>304</td>
</tr>
<tr>
<td>SYSOUT data set</td>
<td>304</td>
</tr>
<tr>
<td>SYSPUNCH data set</td>
<td>305</td>
</tr>
<tr>
<td>DBDSRC data set</td>
<td>308</td>
</tr>
<tr>
<td>PSBSRC data set</td>
<td>308</td>
</tr>
<tr>
<td>SYSPRINT data set</td>
<td>308</td>
</tr>
<tr>
<td>MAPOUT data set</td>
<td>325</td>
</tr>
<tr>
<td>OPTPRT data set</td>
<td>325</td>
</tr>
<tr>
<td>DBD/PSB/ACB Reversal Site Default Generation utility</td>
<td>326</td>
</tr>
<tr>
<td>Reversal Site Default Generation utility overview</td>
<td>326</td>
</tr>
<tr>
<td>Setting site default values for the DBD/PSB/ACB Reversal utility</td>
<td>326</td>
</tr>
<tr>
<td>JCL requirements for the Reversal Site Default Generation utility</td>
<td>329</td>
</tr>
<tr>
<td>Control statements for the Reversal Site Default Generation utility</td>
<td>330</td>
</tr>
<tr>
<td>Output from the DBD/PSB/ACB Reversal Site Default Generation utility</td>
<td>331</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 9. Catalog Manager utility</th>
<th>333</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog Manager utility overview</td>
<td>334</td>
</tr>
</tbody>
</table>
About this information

IBM® IMS™ Library Integrity Utilities for z/OS® (also referred to as IMS Library Integrity Utilities or IMS LIU) is a tool that helps you in managing data for the libraries, such as the DBD libraries, PSB libraries, ACB libraries, and RECON data sets that you use when referring to the IMS database.

These topics are designed for system programmers, application programmers, system analysts, database administrators, and computer operators perform these tasks:

- Understand the functions and utilities of IMS Library Integrity Utilities
- Run and use IMS Library Integrity Utilities after it is installed
- Interpret IMS Library Integrity Utilities reports
- Diagnose and recover from IMS Library Integrity Utilities problems

To use these topics, you should have a working knowledge of:

- The z/OS operating system
- ISPF
- SMP/E

Always refer to the IMS Tools Product Documentation web page for complete product documentation resources:


The IMS Tools Product Documentation web page includes:

- Links to [IBM Knowledge Center](http://www.ibm.com/support/docview.wss?uid=swg27020942) for the user guides ("HTML")
- PDF versions of the user guides ("PDF")
- Program Directories for IMS Tools products
- Recent updates to the user guides, referred to as "Tech docs" ("See updates to this book!")
- Technical notes from IBM Software Support, referred to as "Tech notes"
- White papers that describe product business scenarios and solutions
IBM IMS Library Integrity Utilities for z/OS (also referred to as IMS Library Integrity Utilities or IMS LIU) aids you in managing data for the libraries, such as the DBD libraries, PSB libraries, ACB libraries, and RECON data sets that you use when referring to IMS databases.

Topics:
- “What’s new in IMS Library Integrity Utilities” on page 2
- “IMS Library Integrity Utilities terminology” on page 6
- “What does IMS Library Integrity Utilities do?” on page 8
- “IMS Library Integrity Utilities solutions” on page 13
- “Functional enhancements in IMS Library Integrity Utilities Version 2” on page 15
- “Service updates and support information” on page 20
- “Product documentation and updates” on page 21
- “Accessibility features for IMS Library Integrity Utilities” on page 23
What's new in IMS Library Integrity Utilities

This topic summarizes the technical changes for this edition.

New and changed information is indicated by a vertical bar (|) to the left of a change. Editorial changes that have no technical significance are not noted.

SC19-3979-03 (September 2017)

DBD/PSB/ACB Compare utility

- The utility supports a new parameter, METADATA, for the NOCOMP statement (APAR PI67745). This parameter indicates that the metadata fields in DBD, PSB, or ACB are not compared. The behavior of this parameter is the same as NOCOMP=CATALOG. For more information, see “NOCOMP control statement” on page 213.
- The utility supports new parameters for the NOCOMP statement (APAR PI71280). The new parameters are LANG, LIST, PROCLOPT, PROCSQ, PROCEOED, PSB_ACCESS, and PSB_PSELOPT. For more information, see “NOCOMP control statement” on page 213.

IMS Administration Tool support

IMS Library Integrity Utilities extends the functions of IBM IMS Administration Tool for z/OS (APAR PI67745). If IMS Library Integrity Utilities is configured for IMS Administration Tool, IMS Library Integrity Utilities enables the following functions of IMS Administration Tool:

- View IMS databases (DBDs) and program views (PSBs) of Database and Application administration
- IMS Catalog Management
- Program View of IMS SPUFI

For more information, see the IMS Administration Tool User’s Guide and Reference.

Explanations of return codes and messages that you might receive using IMS Library Integrity Utilities under IMS Administration Tool are provided in Chapter 14, “Troubleshooting,” on page 437.

SC19-3979-02 (March 2016)

DBD/PSB/ACB Compare utility

The utility supports a new control statement, CTLSTMT. Use this control statement to echo the SYSIN control statements and selected runtime options to the SYSPRINT data set (APAR PI35148). For more information, see “Control statements for the DBD/PSB/ACB Compare utility” on page 207.

DBD/PSB/ACB Mapper utility

The utility prints the version of IMS that was used to generate the control block in the SYSPRINT data set (APAR PI33159). For more information, see the reports in “SYSPRINT data set” on page 259.

DBD/PSB/ACB Reversal utility

- The following new control statements are supported (APAR PI47105):
  - ACB_REFENRECED option to decode DBD-type ACBs that are not referenced by any PSB-type ACBs.
  - PGM_COBOL option to set LANG=COBOL instead of LANG=ASSEM in the decoded PSB source.
– DECOPT PCB_LABEL option to print the PCBNAME in the label instead of printing the PCBNAME parameter in the decoded PSB source.

– DECOPT SENSEG_PROCOPT option to always print the SENSEG PROCOPT value in the decoded PSB source.

This APAR also adds a new function control statement, UNREF ACB. Use this control statement to generate the Unreferenced ACB(DBD) report in the SYSPRINT data set.

For more information, see the following topics:

– “Control statements for the DBD/PSB/ACB Reversal utility” on page 288

– “Unreferenced ACB(DBD) report” on page 323

• LISTLIB DBD and LISTLIB PSB control statements generate DBD and PSB library member list reports in the SYSPRINT data set (APAR PI53117). The report contains information about the members in the data sets that are concatenated to DBDLIB DD or PSBLIB DD. For more information, see the following topics:

  – “Control statements for the DBD/PSB/ACB Reversal utility” on page 288

  – “Library member list report for DBD or PSB” on page 323

Catalog Manager utility

The utility provides the function to check the time stamp of DBD-type ACB and PSB-type ACB members in the IMS catalog directory (APAR PI36550). For more information, see “Catalog Manager utility overview” on page 334.

DBD/PSB Map Viewer

The following capabilities have been added to the DBD/PSB Map Viewer for Management Console (APAR PI50732):

• Viewing program specifications, PSB source, and PSB XML document

• Listing logical DBDs and PSBs that refer to a specific DBD

SC19-3979-01

Integrity Checker utility

Instructions to maintain RDEs for HALDBs and DEDBs during an online database change were added. For more information, see the following topics:

• “Considerations for activating Integrity Checker” on page 50

• “Altering the definition of a DMB verification-enabled online HALDB by using the HALDB alter function” on page 72

• “Altering the definition of a DMB verification-enabled online DEDB by using the DEDB Alter utility (DBFUDA00)” on page 73

Multiple Resource Checker utility

The utility supports a new option that you can use when you compare multiple sets of RECON data sets. When you use this option, you can compare only the database definitions in RECON data sets or both database definitions and recovery environment definitions in RECON data sets (APAR PI12851). The following topics were added or updated to support this enhancement:

• “Multiple Resource Checker utility overview” on page 158
DBD/PSB/ACB Compare utility

- The utility supports new parameters for the NOCOMP statement. The new parameters are the AREA, RMNAME, COMPRTN, PCBNAME, and KEYLEN parameters (APAR PI21707). The following topics were updated to support this enhancement:
  - “Restrictions and considerations for the DBD/PSB/ACB Compare utility” on page 202
  - “Control statements for the DBD/PSB/ACB Compare utility” on page 207

- The utility can generate source-level compare reports even when no difference is found (APAR PI18270). The following topics were added or updated to support this enhancement:
  - “DBD/PSB/ACB Compare utility overview” on page 200
  - “Restrictions and considerations for the DBD/PSB/ACB Compare utility” on page 202
  - “Control statements for the DBD/PSB/ACB Compare utility” on page 207

DBD/PSB/ACB Reversal utility

- The utility prints the following information with the converted IMS DBDGEN or IMS PSBGEN utility control statements in the SYSPUNCH data set (APAR PI21708):
  - The date and time when the DBD, PSB, or ACB control block was converted to IMS DBDGEN or IMS PSBGEN utility control statements
  - The version of IMS that was used to create the control block

The report examples were updated in “SYSPUNCH data set” on page 305.

- The utility supports a new keyword, POPTREF, for defining criteria to identify and report on specific PSBs (APAR PI17798). The following topics were added or updated to support this enhancement:
  - “DBD/PSB/ACB Reversal utility overview” on page 280
  - “Control statements for the DBD/PSB/ACB Reversal utility” on page 288
  - “PSB PROCOPT reference reports for PSB and ACB(PSB)” on page 322

Catalog Manager utility

The new Catalog Manager utility can help you ensure that DBDs and PSBs in the IMS catalog match the DBD and PSB members in the ACB libraries (APAR PI21200). For more information, see Chapter 9, “Catalog Manager utility,” on page 333.
ACBLIB Analyzer utility
The utility supports a new operand, GENDATE, for the LISTLIB command. This operand specifies to print the date and time when the ACB members were generated in the Library Contents report (APAR PI09309). The following topics were added or updated to support this enhancement:
• “JCL requirements for the ACBLIB Analyzer utility” on page 382
• “ACBSYSIN control statements” on page 384
• “Output from the ACBLIB Analyzer utility” on page 386
• “Library Contents report” on page 388

Also see “Functional enhancements in Version 2 Release 2” on page 15 for more information about these enhancements.

SC19-3979-00
Integrity Checker utility
Instructions to activate Integrity Checker are revised. To activate Integrity Checker, follow the instructions from “Planning for Integrity Checker configuration” on page 39.

Multiple Resource Checker utility
A new utility, Multiple Resource Checker, is added. You can use this utility to check the consistency of DBD and PSB definitions in the RECON data sets, DBD, PSB, and ACB libraries and to report differences between RECON data sets. See Chapter 5, “Multiple Resource Checker utility,” on page 157.

DBD/PSB/ACB Compare utility
DBD/PSB/ACB Compare can generate a source-level compare report for ACBLIB members that are generated by different IMS releases. For more information, see Chapter 6, “DBD/PSB/ACB Compare utility,” on page 199.

Others
Supports the DBD Map Viewer, which is the IMS Library Integrity Utilities extension for Management Console and Tools Base Administration Console, to view the graphical visualization of a database structure map, the DBD macro source, and the DBD XML document. For more information, see “Functional enhancements in Version 2 Release 2” on page 15.
IMS Library Integrity Utilities terminology

IMS Library Integrity Utilities information includes several unique terms that you need to understand before you begin to use IMS Library Integrity Utilities.

To make this information easier to read, the version and release levels of IMS are abbreviated, as follows:
- **IMS Version 15** refers to IMS Version 15 Release 1.

The various versions of IMS are referred to simply as IMS, except where distinctions among them must be made.

The following table summarizes the terminology used in this information.

*Table 1. IMS Library Integrity Utilities terminology*

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACB</td>
<td>Application control block</td>
</tr>
<tr>
<td>BPE</td>
<td>Base Primitive Environment</td>
</tr>
<tr>
<td>DBD</td>
<td>Database description</td>
</tr>
<tr>
<td>DBRC</td>
<td>IMS Database Recovery Control facility</td>
</tr>
<tr>
<td>DIF</td>
<td>Device input format</td>
</tr>
<tr>
<td>DOF</td>
<td>Device output format</td>
</tr>
<tr>
<td>DMB</td>
<td>Data management block</td>
</tr>
<tr>
<td>DMCB</td>
<td>DEDB master control block</td>
</tr>
<tr>
<td>DRD</td>
<td>Dynamic resource definition</td>
</tr>
<tr>
<td>LICON</td>
<td>IMS LIU Integrity Control</td>
</tr>
<tr>
<td>MFS</td>
<td>Message Format Services</td>
</tr>
<tr>
<td>MID</td>
<td>Message input descriptor</td>
</tr>
<tr>
<td>MOD</td>
<td>Message output descriptor</td>
</tr>
<tr>
<td>PSB</td>
<td>Program specification block</td>
</tr>
<tr>
<td>RDE</td>
<td>Registered DMB entry</td>
</tr>
<tr>
<td>RDDS</td>
<td>Resource definition data set</td>
</tr>
<tr>
<td>RECON</td>
<td>Recovery Control</td>
</tr>
</tbody>
</table>

In this information, the following abbreviations are used for product and component names.

*Table 2. Product short names*

<table>
<thead>
<tr>
<th>Short name</th>
<th>Product name</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS Administration Tool</td>
<td>IBM IMS Administration Tool for z/OS</td>
</tr>
<tr>
<td>IMS Database Recovery Facility</td>
<td>IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility</td>
</tr>
<tr>
<td>IMS Database Reorganization Expert</td>
<td>IBM IMS Database Reorganization Expert for z/OS</td>
</tr>
<tr>
<td>Short name</td>
<td>Product name</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IMS HP Fast Path Utilities</td>
<td>IBM IMS Fast Path Solution Pack for z/OS: IMS High Performance Fast Path Utilities</td>
</tr>
<tr>
<td>IMS HP Image Copy</td>
<td>IBM IMS High Performance Image Copy for z/OS</td>
</tr>
<tr>
<td>IMS HP Load</td>
<td>IBM IMS High Performance Load for z/OS</td>
</tr>
<tr>
<td>IMS Library Integrity Utilities or IMS LIU</td>
<td>IBM IMS Library Integrity Utilities for z/OS (this product)</td>
</tr>
<tr>
<td>IMS Online Reorganization Facility</td>
<td>IBM IMS Database Solution Pack for z/OS: IMS Online Reorganization Facility</td>
</tr>
<tr>
<td>Management Console</td>
<td>IBM Management Console for IMS and DB2® for z/OS</td>
</tr>
</tbody>
</table>
What does IMS Library Integrity Utilities do?

IMS Library Integrity Utilities aids you in managing data for the libraries, such as the DBD libraries, PSB libraries, ACB libraries, and RECON data sets that you use when referring to IMS databases.

Some typical data management functions are:
- To prevent the database corruption that the use of an incorrect member of a library can cause.
- To check the consistency among each library.
- To check, compare, change, generate, and maintain the members of a library.

These functions are provided by the following utilities:
- “Integrity Checker utility”
- “Consistency Checker utility” on page 9
- “Multiple Resource Checker utility” on page 10
- “DBD/PSB/ACB Compare utility” on page 10
- “DBD/PSB/ACB Mapper utility” on page 10
- “DBD/PSB/ACB Reversal utility” on page 10
- “Catalog Manager utility” on page 11
- “Advanced ACB Generator utility” on page 11
- “ACBLIB Analyzer utility” on page 12
- “MFS Reversal utility” on page 12
- “MFS Compare utility” on page 12

Note: The utilities of IMS Library Integrity Utilities do not support IMS Partition DB (5697-A06, 5697-D85) or any other products with equivalent functions.

Integrity Checker utility

If the control blocks that IMS uses for access to a database are not the same as the ones that IMS used to load the database, data integrity can be compromised. This condition is one of the most common causes of corruption in IMS databases. Some typical reasons for using a wrong control block are as follows:
- A batch program uses a test DBD library to update a production database.
- A batch program uses a new DBD before the associated database is reorganized for that DBD.
- An online IMS subsystem uses an old ACB to update a database that has been reorganized since its associated DBD was changed.

Using a wrong IMS control block is a common procedural error, which IMS or DBRC cannot prevent.

Integrity Checker addresses this error. To prevent both batch programs and IMS systems from using the wrong IMS control blocks for access to a database, it verifies the following two IMS control blocks during the database authorization:
- The DMB that was used to load the database
- The DMB that IMS is using to get access to a database

In a batch environment, your program uses DBDs to get the database definition needed for access to a database. In an online environment, your IMS subsystem
uses ACBs to get that database definition. In either case, DL/I internally builds a control block, called DMB, for each database to be accessed, from a DBD or an ACB. As soon as DMB is created, all information about the database definition refers to the DMB, not to the DBD or the ACB. Integrity Checker verifies the internal control block DMB rather than the DBD or ACB itself.

If a mismatch is found, Integrity Checker denies authorization to the database. This mechanism prevents any accidental updating of a database with an incorrect DBD, which would result in corruption of the database.

Some benefits of using Integrity Checker are as follows:

- It prevents the risk of corrupting data by using the wrong DBD.
- It prevents the risk that corrupted data will cause system outages.
- It reduces the cost of recovering databases that have been corrupted.

Integrity Checker provides an option to record database accesses that are made for database update, load, and unload operations. When you enable this option, Integrity Checker records database accesses that are made from the utilities and the application programs that are supported by Integrity Checker.

**Consistency Checker utility**

Consistency Checker ensures that the necessary definitions in an IMS subsystem have been created for your database or your application program. For a DBD in the DBD library, Consistency Checker verifies whether the following definitions have been created correctly in each library and whether these definitions are consistent with the DBD:

- The ACB in the ACB library
- The database definition entry in the MODBLKS module
- The database definition entry in the resource definition data sets (RDDSs)
- The DFSMDA dynamic allocation member for database data set in the MDA library
- The DB and DSG registration record in the RECON

Consistency Checker decides which definitions are to be verified depending on the user input and the database organization defined in the specified DBD.

**Related reading:** For the libraries to be verified for each database organization, see Table 4 on page 126.

For a PSB in the PSB library, Consistency Checker verifies whether the following definitions have been created correctly in each library and whether these definitions are consistent with the PSB:

- The ACB in the ACB library
- The application program definition entry in the MODBLKS module
- The application program definition entry in the resource definition data sets (RDDSs)

Consistency Checker generates reports after checks and helps you determine which definitions are needed before you start an IMS subsystem.
**Multiple Resource Checker utility**

The Multiple Resource Checker utility checks the consistency across multiple resources.

For DBDs, Multiple Resource Checker verifies whether the following resources exist in each library and whether the definitions contained are consistent:

- DBD members in DBD libraries
- ACB members in ACB libraries
- DB and DBDS records registered in RECON data sets

For PSBs, Multiple Resource Checker verifies whether the following definitions are the same in each library:

- PSB members in PSB libraries
- ACB members in ACB libraries

Multiple Resource Checker can process up to 10 DBD libraries, PSB libraries, ACB libraries, and 10 sets of RECON data sets in one job. The utility generates a Resource Check Summary report, which contains a matrix table that summarizes the results of checking.

You can use the Multiple Resource Checker to ensure that IMS resources are the same and, if inconsistencies are found, investigate which resources are inconsistent.

**DBD/PSB/ACB Compare utility**

The DBD/PSB/ACB Compare utility reports the differences between database description (DBD) control blocks, program specification blocks (PSB), or application control blocks (ACB). By using this utility, you can compare control blocks that have the same name but that reside in different libraries, or control blocks that have different names and that reside in the same object library or in different object libraries.

The reports produced by DBD/PSB/ACB Compare enables you to check the differences between DBDs, PSBs, and ACBs.

**DBD/PSB/ACB Mapper utility**

The DBD/PSB/ACB Mapper utility produces printed maps (pictures of the segment hierarchy) from DBDs, PSBs, and ACBs. It also produces detailed reports that describe DBDs, PSBs, and ACBs.

The maps produced by DBD/PSB/ACB Mapper can be used as recording mediums to retain the historical and current status of the IMS databases. They can also be used as a reference in comparing and evaluating the database requirements of current and proposed applications.

**DBD/PSB/ACB Reversal utility**

The DBD/PSB/ACB Reversal utility converts the DBD/PSB/ACB control blocks back into IMS DBDGEN/PSBGEN utility control statements. The DBD/PSB/ACB Reversal utility is helpful if you have lost your source libraries that contain DBDGEN/PSBGEN utility control statements.
The DBD/PSB/ACB Reversal utility also produces useful summary reports of IMS DBD, PSB, and ACB libraries. These reports represent the IMS member information, such as DBD, PSB, and ACB organization, PCB PROCOPT, and the relations among members.

**Catalog Manager utility**

The Catalog Manager utility helps you ensure that the DBDs and PSBs in the IMS catalog are maintained correctly. It validates that the DBDs and PSBs in the ACB libraries match the DBDs and PSBs in the IMS catalog. The utility generates several reports, including the IMS Catalog Validation report, which contains the results of the validation process.

**Advanced ACB Generator utility**

Advanced Application Control Block Generator (also referred to as Advanced ACBGEN utility) is a functional replacement for the IMS Application Control Blocks Maintenance utility (DFSUACB0), also referred to as IMS ACBGEN utility), with improvements. The Advanced ACBGEN utility can also replace the IMS ACBGEN utility (DFSUACB0) to generate ACB members within ACB Generation and Catalog Populate utility (DFS3UACB) jobs.

The Advanced ACBGEN utility uses some of the IMS provided modules and replaces others. Additionally, several utilities are provided to display and audit the contents of an ACB library.

Advanced ACBGEN utility provides the following features:

- **Extensive and informative set of reports**
  - Data set information about the ACB, PSB, and DBD libraries used
  - Summary of PSBs added, replaced, and deleted during ACBGEN
  - Summary of DBDs added, replaced, not replaced, and deleted during ACBGEN
  - Distribution of PSB and DBD sizes
  - Summary of DFSnnnn messages issued

- **Utility to analyze an ACB library for potential problems**
  - Verifies that all members are at the same IMS version and release level
  - Verifies that the ACB library does not contain any PSBs or DBDs inadvertently placed there during a DBDGEN or a PSBGEN
  - Generates a distribution of unique ACBGEN dates

- **Improved performance**
  - Elapsed time reductions of 200% to 500% (2 - 5 times as fast)

- **Reduced use of system resources**
  - Reduces use of CPU
  - Reduces EXCPs
  - Reduces volume of SYSPRINT

- **Easy to use**
  - Implemented by adding the load library of IMS Library Integrity Utilities to the top of the STEPLIB DD concatenation when invoking current JCL procedures
  - Uses but does not modify IMS modules
ACBLIB Analyzer utility

The ACBLIB Analyzer utility verifies that all ACB library members are at the same IMS Version/Release level, and that all members were placed in the ACB library by the ACBGEN process—that is, the library was not inadvertently used during a DBDGEN or PSBGEN. The utility program also produces several reports.

MFS Reversal utility

MFS Reversal is designed to convert Message Format Services (MFS) control blocks (MIDs, MODs, DIFs, and DOFs) back into IMS MFS utility control statements.

Note: MID refers to Message Input Descriptor and MOD to Message Output Descriptor; DIF refers to Device Input Format, and DOF to Device Output Format.

The primary purpose of MFS Reversal is to re-create MFS sources. This recovery function is important if you have lost your MFS source library or suspect a difference between the control blocks being used and the source.

In addition to decoding MFS control blocks, MFS Reversal provides valuable summary reports of IMS format libraries. You can use these reports to obtain the cross-reference information between the MID/MODs and the DIF/DOFs, and the cross-reference information between the MIDs and the MODs. Also, if a MID or MOD has been selected for reversal, the report informs you of other MIDs and MODs names that were not selected for reversal, but refer to the same DIFs and DOFs referenced by the selected MID or MOD.

You can optionally request the MFS Reversal utility to copy the selected MFS control blocks and their associated control blocks from the IMS format library to a user-specified partitioned data set.

MFS Compare utility

MFS Compare is designed to compare two sets of MFS format control blocks from two MFS format libraries to quickly highlight differences between them.

MFS Compare enables you to compare two MFS format libraries. If you are not sure whether the MFS you are running corresponds to your MFS source library, you can create MFS control blocks from your source statements and then compare the control blocks with the control blocks you are running. Thus, you can use MFS Compare to validate what actually is operating in a particular IMS environment.
IMS Library Integrity Utilities solutions

IMS Library Integrity Utilities helps you to validate, compare, report, and recover IMS libraries.

It is recommended that you run Integrity Checker in everyday operations. If the utility is activated, it ensures, every time IMS gets access to databases, that the ACB or DBD is the correct one by comparing the contents of the DMB and the DMB information registered in the LICON data set. It can be activated either online or by batch. If any inconsistency exists, it must be resolved, and to resolve the inconsistency, other utilities that are introduced in this topic can be used.

Consistency Checker can be used, for example, when migrating your IMS system, or creating new IMS subsystems. By use of Consistency Checker, you can check the necessary DBD, PSB, or both for your new system, and you can make sure that the associated definitions that you have made, such as the following definitions, are consistent with the DBD or the PSB:

For DBDs:
- The ACB in the ACB library
- The database definition entry in the MODBLKS module
- The database definition entry in the resource definition data sets (RDDSs)
- The DFSMDA dynamic allocation member for the database data set in the MDA library
- The DB and DSG registration record in the RECON

For PSBs:
- The ACB in the ACB library
- The application program definition entry in the MODBLKS module
- The application program definition entry in the resource definition data sets (RDDSs)

Multiple Resource Checker can be used, for example, to check the consistency of the IMS resources between the test environment and the production environment. By using Multiple Resource Checker, you can check the consistency of the following resources and definitions across multiple libraries and RECON data sets.

For DBDs:
- DBD members across multiple DBD libraries
- ACB members across multiple ACB libraries
- DB and DBDS records registered in RECON data sets

For PSBs:
- PSB members across multiple PSB libraries
- ACB members across multiple ACB libraries

If errors are detected by Integrity Checker, Consistency Checker, or Multiple Resource Checker, the following utilities can support you:
- DBD/PSB/ACB Compare reports the differences between DBDs, PSBs, or ACBs that have the same name but that reside in different libraries.
- DBD/PSB/ACB Mapper produces pictorial layouts of IMS physical and logical databases.
• DBD/PSB/ACB Reversal converts the DBD/PSB control blocks back into IMS
  DBDGEN/PSBGEN utility control statements.
• Advanced Application Control Block Generator provides a high-speed
generation process, which becomes more important when processing large
volumes of IMS ACBs.
Functional enhancements in IMS Library Integrity Utilities Version 2

This topic describes the major functional enhancements that have been made in IMS Library Integrity Utilities Version 2.

Functional enhancements in Version 2 Release 2

This topic describes the major functional enhancements that are made to IMS Library Integrity Utilities Version 2 Release 2.

Subsections:
- “Multiple Resource Checker utility” on page 157
- “DBD/PSB/ACB Compare utility” on page 199
- “DBD/PSB/ACB Mapper utility” on page 16
- “DBD/PSB/ACB Reversal utility” on page 16
- “Catalog Manager utility” on page 16
- “ACBLIB Analyzer utility” on page 16
- “Other enhancements” on page 17

Multiple Resource Checker utility

- This new utility checks the consistency of DBD and PSB definitions in the RECON data sets, DBD, PSB, and ACB libraries. The utility can process up to 10 DBD libraries, PSB libraries, ACB libraries, and 10 sets of RECON data sets in one job. The utility generates a report that contains a matrix table that summarizes the results of consistency check. Additionally, the Multiple Resource Checker utility supports an option that can be used to report differences between two sets of RECON data sets.
- APAR PI12851 adds a new option that is used when comparing multiple sets of RECON data sets. When you use this option, you can compare only the database definitions in RECON data sets or both database definitions and recovery environment definitions in RECON data sets.

For more information, see Chapter 5, “Multiple Resource Checker utility,” on page 157.

DBD/PSB/ACB Compare utility

- DBD/PSB/ACB Compare can generate a source-level compare report for ACBLIB members that are generated by different IMS releases.
- APAR PI18270 provides an option to generate source-level compare reports even when no difference is found.
- APAR PI21707 adds new parameters for the NOCOMP statement. The new parameters are the AREA, RMNAME, COMPRTN, PCBNAME, and KEYLEN parameters.
- APAR PI35148 adds a new keyword, CTLSTMT, for echoing the SYSIN control statements and selected runtime options to the SYSPRINT data set.
- APAR PI71280 adds new parameters for the NOCOMP statement. The new parameters are LANG, LIST, PROCOPT, PROCSEQ, PROCSEQD, PSB_ACCESS, and PSB_PSELOPT.

For more information, see Chapter 6, “DBD/PSB/ACB Compare utility,” on page 199.
DBD/PSB/ACB Mapper utility

APAR PI33159 enhances the utility to print the version of IMS that was used to generate the control block in the SYSPRINT data set.

For more information, see Chapter 7, “DBD/PSB/ACB Mapper utility,” on page 243.

DBD/PSB/ACB Reversal utility

- APAR PI17798 adds a new keyword, POPTREF, for defining criteria to identify and report on specific PSBs.
- APAR PI21708 enhances the utility to print the following information with the converted IMS DBDGEN or IMS PSBGEN utility control statements in the SYSPUNCH data set:
  - The date and time when the DBD, PSB, or ACB control block was converted to IMS DBDGEN or IMS PSBGEN utility control statements
  - The version of IMS that was used to create the control block
- APAR PI47105 provides the following new options:
  - ACB_REFERENCED option to decode DBD-type ACBs that are not referenced by any PSB-type ACBs.
  - PGM_COBOL option to set LANG=COBOL instead of LANG=ASSEM in the decoded PSB source.
  - DECOPT PCB_LABEL option to print the PCBNAME in the label instead of printing the PCBNAME parameter in the decoded PSB source.
  - DECOPT SENSEG_PROCOPT option to always print the SENSEG PROCOPT value in the decoded PSB source.

The APAR also adds a new function control statement, UNREF ACB, which generates an Unreferenced ACB(DBD) report. The report contains information about DBD-type ACBs that are not referenced by any PSB-type ACBs in the ACBLIB.

- APAR PI53117 adds new control statements, LISTLIB DBD and LISTLIB PSB, for generating DBD or PSB library member list report.

For more information, see Chapter 8, “DBD/PSB/ACB Reversal utility,” on page 279.

Catalog Manager utility

- APAR PI21200 provides a new utility, the Catalog Manager utility. This new utility helps you ensure that DBDs and PSBs in the IMS catalog match the DBD and PSB members in the ACB libraries.
- APAR PI36550 provides the function to check the time stamp of DBD-type ACB and PSB-type ACB members in the IMS catalog directory.

For more information, see Chapter 9, “Catalog Manager utility,” on page 333.

ACBLIB Analyzer utility

APAR PI09309 adds a new operand, GENDATE=YES, for the LISTLIB command of the ACBSYSIN control statement. If you specify the new GENDATE=YES operand, the Library Contents report additionally includes the date and time when the ACB members were generated. For more information, see “Library Contents report” on page 388.
Other enhancements

Provides the DBD/PSB Map Viewer, which is the IMS Library Integrity Utilities extension for Management Console and Tools Base Administration Console. You can display the graphical visualization of a database structure map, program specifications, DBD and PSB macro source, and DBD and PSB XML document in the web interface of Management Console. You can also use the cross reference feature to list logical DBDs and PSBs that refer to a specific DBD.

To use this feature, IMS Database Solution Pack or IMS Fast Path Solution Pack is required. For more information, see the topic "Setting up the DBD/PSB Map Viewer for Management Console" in IMS Database Solution Pack: Overview and Customization or in IMS Fast Path Solution Pack: Overview and Customization.

Functional enhancements in Version 2 Release 1

This topic describes the major functional enhancements that are made to IMS Library Integrity Utilities Version 2 Release 1.

Subsections:

- “Integrity Checker utility”
- "DBD/PSB/ACB Compare utility“ on page 18
- "DBD/PSB/ACB Reversal utility” on page 18
- “Advanced ACB Generator utility” on page 18
- “MFS Reversal utility” on page 19

Integrity Checker utility

Automatic determination of the correct RDE

When you run a time stamp recovery job by using the recovery function of IMS HP Image Copy or IMS Database Recovery Facility, Integrity Checker automatically recovers the RDE that was valid at the specified recovery time, and uses that RDE for DMB verification. To use this automatic determination function, you must specify a large value for the GENMAX option so that enough number of expired RDEs can be kept in the LICON data set.

Note: If IMS HP Image Copy Version 4 Release 1 is used, APAR PK69500 must be applied to use the automatic determination function.

Enhancement for the verification option

You can set the CHECKRV verification option for database recovery jobs by specifying the CHECKRV option in global option modules or in RDEs.

You can verify the changes in the DEDB partition selection exit routines by specifying the CHKFPSEL option in the global option modules or in the RDEs. This option is provided by APARs PM37150 and PM46494.

For more information, see “JCL requirements for the FABLPGEN program” on page 92 and "INIT.DB command” on page 101.

Creating RDEs under IMS Database Reorganization Expert

When you reorganize a database by using the Smart Reorg utility of IMS Database Reorganization Expert or the IPR Driver of IMS Parallel Reorganization that has APAR PK69458 applied, and you specify the NEWDBD= control statement, Integrity Checker creates a new RDE for the new DBD.
Recording database accesses

Integrity Checker can record database accesses when database update, load, and unload operations are made by the supported utilities or application programs. The recorded access information includes the subsystem name and the time of the access. To use this feature, apply the following APARs:

- APAR PM17661, PM21355, PM28084, and PM30910 to IMS Library Integrity Utilities
- APAR PM27942, if IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities is used
- APAR PK79386, if IMS High Performance Fast Path Utilities Version 3 Release 2 is used

Support for global option modules prefixed with LIUG

Integrity Checker supports global option modules that are prefixed with LIUG. You can use LIUGINST or LIUGimsid in addition to LIU@INST and LIU@imsid. You must apply APAR PM30256 to use the global option modules that are prefixed with LIUG.

Enhancement to the LICON utility to support HALDB partitions

The LICON utility of Integrity Checker supports creating and verifying RDEs for HALDB partitions. You can use the INIT.DB command to create RDEs for HALDB partitions and the VERIFY.DB command to verify the RDEs of HALDB partitions.

DBD/PSB/ACB Compare utility

The DBD/PSB/ACB Compare can compare a DBD to its corresponding ACB, and a PSB to its corresponding ACB. For more information, see Chapter 6, “DBD/PSB/ACB Compare utility,” on page 199.

The DBD/PSB/ACB Compare utility can compare members that have different names and that reside in the same object library or in different object libraries. This feature is provided by APAR PM10930.

DBD/PSB/ACB Reversal utility

The DBD/PSB/ACB Reversal Site Default Generation utility can be used to set user-defined default values for the SYSIN control statements of DBD/PSB/ACB Reversal. This feature is provided by APAR PM91313. For more information, see “DBD/PSB/ACB Reversal Site Default Generation utility” on page 326.

Advanced ACB Generator utility

If an abend, such as a system B37 abend, occurs on an ACBLIB data set during the BUILD DBD= process for changed DBDs, and you have corrected the space problem, the Advanced ACBGEN utility processes, during the rerun of the job, the DBDs that are specified with BUILD DBD= control statements and all the necessary PSBs. Therefore, you no longer need to run BUILD PSB= or BUILD PSB=ALL to force the rebuild of PSBs that are affected by the changes in the DBDs.

Support for IMS ACBGEN enhancements for IMS catalog

The Advanced ACBGEN utility can replace the IMS ACBGEN utility (DFSUACB0) to generate ACB members within ACB Generation and Catalog Populate utility (DFS3UACB) jobs. For more information, see “Advanced ACBGEN utility overview” on page 350.
The Advanced ACBGEN utility also supports the IMS catalog feature that has been added to the IMS ACBGEN utility. When an ACBCATWK DD statement is specified for an IMS ACBGEN utility job, the IMS ACBGEN utility generates a list of the ACB members in the data set that is specified by the ACBCATWK DD statement. If you specify an ACBCATWK DD for the Advanced ACBGEN utility job, the Advanced ACBGEN utility also generates a list of the ACB members in the data set that is specified by the ACBCATWK DD. For more information, see “JCL requirements for the Advanced ACBGEN utility” on page 355.

MFS Reversal utility

Enhancement of the MID/MOD XREF report
The MID/MOD XREF report has been enhanced to display the cross-reference information between MIDs and MODs. This feature is provided by APAR PK84633.

Copy function for MFS format library members
The MFS Reversal utility can copy the selected members and their associated members (MIDs, MODs, DIFs, and DOFs) from the IMS format library to a user-specified partitioned data set. You can enable the copy function by the OPTION control statement. This feature is provided by APAR PM03227.
Service updates and support information

Service updates and support information for this product, including software fix packs, PTFs, frequently asked questions (FAQs), technical notes, troubleshooting information, and downloads, are available from the web.

To find service updates and support information, see the following website:

Product documentation and updates

IMS Tools information is available at multiple places on the web. You can receive updates to IMS Tools information automatically by registering with the IBM My Notifications service.

Information on the web

Always refer to the IMS Tools Product Documentation web page for complete product documentation resources:


The IMS Tools Product Documentation web page includes:
- Links to IBM Knowledge Center for the user guides ("HTML")
- PDF versions of the user guides ("PDF")
- Program Directories for IMS Tools products
- Recent updates to the user guides, referred to as "Tech docs" ("See updates to this book!")
- Technical notes from IBM Software Support, referred to as "Tech notes"
- White papers that describe product business scenarios and solutions

IBM Redbooks® publications that cover IMS Tools are available from the following web page:

http://www.redbooks.ibm.com

The IBM Information Management System website shows how IT organizations can maximize their investment in IMS databases while staying ahead of today's top data management challenges:

https://www.ibm.com/software/data/ims/

Receiving documentation updates automatically

To automatically receive emails that notify you when new technote documents are released, when existing product documentation is updated, and when new product documentation is available, you can register with the IBM My Notifications service. You can customize the service so that you receive information about only those IBM products that you specify.

To register with the My Notifications service:
1. Go to http://www.ibm.com/support/mysupport
2. Enter your IBM ID and password, or create one by clicking register now.
3. When the My Notifications page is displayed, click Subscribe to select those products that you want to receive information updates about. The IMS Tools option is located under Software > Information Management.
4. Click Continue to specify the types of updates that you want to receive.
5. Click Submit to save your profile.
How to send your comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other IBM product documentation, use one of the following options:

- Use the IBM Knowledge Center Contact Us link.
- Use the online reader comment form, which is located at http://www.ibm.com/software/data/rcf/.
- Send your comments by email to comments@us.ibm.com. Include the name of the book, the part number of the book, the version of the product that you are using, and, if applicable, the specific location of the text you are commenting on, for example, a page number or table number.

Prerequisite knowledge

Before using the information, you should understand basic IMS concepts, the IMS environment, and your installation’s IMS system.

The IMS publications are prerequisite for all IMS Library Integrity Utilities components.
Accessibility features for IMS Library Integrity Utilities

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

Accessibility features

The major accessibility feature in IMS Library Integrity Utilities is the keyboard-only operation for ISPF editors. It uses the standard TSO/ISPF interface.

Keyboard navigation

You can access the information center and IMS ISPF panel functions by using a keyboard or keyboard shortcut keys.

For information about navigating the IMS ISPF panels using TSO/E or ISPF, see the following guides:

• z/OS ISPF User’s Guide, Volume 1
• z/OS TSO/E Primer
• z/OS TSO/E User’s Guide

These guides describe how to use ISPF, including the use of keyboard shortcuts or function keys (PF keys), include the default settings for the PF keys, and explain how to modify their functions.

IBM and accessibility

See the IBM Human Ability and Accessibility Center at [www.ibm.com/able](http://www.ibm.com/able) for more information about the commitment that IBM has to accessibility.
Chapter 2. Configuring IMS Library Integrity Utilities

After you install IMS Library Integrity Utilities, you can use the utilities without configuration except for the Integrity Checker utility, the Consistency Checker utility, and the Multiple Resource Checker utility. If you have been using the Integrity Checker utility or the Advanced ACBGEN utility of an earlier release of IMS Library Integrity Utilities, review the migration steps before you install the product.

Topics:
• “Hardware and software prerequisites” on page 26
• “Configuring for initial installation” on page 27
• “Migration procedures” on page 28
Hardware and software prerequisites

Before you install IMS Library Integrity Utilities, prepare an environment that meets the software and hardware requirements of the product.

Complete information about installation requirements, prerequisites, and procedures for IMS Library Integrity Utilities is in the following publications:
- Program Directory for IBM IMS Library Integrity Utilities for z/OS
- Program Directory for IBM IMS Database Solution Pack for z/OS
- Program Directory for IBM IMS Fast Path Solution Pack for z/OS

Hardware

IMS Library Integrity Utilities runs on any hardware configuration that supports the required versions of IMS.

Software

IMS Version 13 Release 1 or later, or IMS Database Value Unit Edition Version 13 Release 1 or later.

Integrity Checker utility
- To activate Integrity Checker, a DBRC environment is required. Also, the DBRC RECON data sets must be generated by IMS Version 11 Release 1 or later releases.
- To migrate your LICON data set from IMS Library Integrity Utilities Version 1 by using the LICON data set migration utility, IBM DFSORT (Data Facility Sort), which is a part of z/OS, or a functionally equivalent sort/merge program, is required.
- RECON data sets that are generated with MINVERS=9.1 or greater, and the IMS SDFSRESL library of IMS Version 9 Release 1 or of a later release can be used.

Multiple Resource Checker utility
- Supports DBDLIBs and PSBLIBs that are generated by any version of IMS. Supports ACBLIBs that are generated by IMS Version 11 or later.
- RECON data sets that are generated by IMS Version 11 or later, and the IMS SDFSRESL library of IMS Version 11 Release 1 or of a later release can be used.

DBD/PSB/ACB Compare utility
DBD/PSB/ACB Mapper utility
DBD/PSB/ACB Reversal utility
- Supports DBDLIBs and PSBLIBs that are generated by any version of IMS. Supports ACBLIBs that are generated by IMS Version 11 or later.

Catalog Manager utility
- Supports ACB libraries that are generated by IMS Version 12 or later.

Advanced ACB Generator utility
- If the LISTLIB command parameter USESORT=YES is specified on the ACBSYSIN control statement of the ACBLIB Analyzer utility, IBM DFSORT (Data Facility Sort), which is a part of z/OS, or a functionally equivalent sort/merge program, is required unless otherwise stated.
Configuring for initial installation

Before you can activate Integrity Checker, Consistency Checker, or Multiple Resource Checker, you must configure the environment. Other utilities require no configuration tasks.

**Integrity Checker**

Configuration tasks include designing an Integrity Checker configuration. Follow the instructions in “Planning for Integrity Checker configuration” on page 39.

**Consistency Checker**

**Multiple Resource Checker**

If you protect DBRC commands or DBRC API requests with RACF®, you must set up the RACF security. Follow the instructions in “Setting up security for Consistency Checker and Multiple Resource Checker.”

If RACF is not used for DBRC commands or DBRC API requests, you can use the utilities without configuration.

**Setting up security for Consistency Checker and Multiple Resource Checker**

Consistency Checker uses the DBRC command utility and Multiple Resource Checker uses the DBRC API to verify the registration of the databases and the data sets to the RECON data sets.

For Consistency Checker, if you protect DBRC commands with RACF by permitting appropriate user access to the profiles, you must permit appropriate users of jobs to use the DBRC command LIST.RECON.

For Multiple Resource Checker, if you protect DBRC API requests with RACF by permitting appropriate user access to the profiles, you must permit appropriate users of jobs to use the following DBRC API requests:

- STARTDBRC
- STOPDBRC
- QUERY

For details about the RACF settings for DBRC command and DBRC API requests, see the topic "DBRC security" in *IMS System Administration*. 
Migration procedures

Out of the utilities provided in IMS Library Integrity Utilities Version 2 Release 2, only the Integrity Checker utility and the Advanced ACBGEN utility require migration tasks.

Other utilities require no migration tasks. You can use other utilities by installing and replacing the IMS Library Integrity Utilities load module library.

Follow the instructions in the following topics to complete migration for the Integrity Checker utility and the Advanced ACBGEN utility:

- "Migrating Integrity Checker"
- "Migrating Advanced ACBGEN" on page 33

Migrating Integrity Checker

The steps to migrate Integrity Checker from an earlier release of the product differ by the version of the product that you have been using and whether BPE-based DBRC was used.

Follow the instructions in the following topics:

- "Migrating from IMS Library Integrity Utilities Version 2 Release 1 (non BPE-based DBRC)"
- "Migrating from IMS Library Integrity Utilities Version 2 Release 1 (BPE-based DBRC)" on page 29
- "Migrating from IMS Library Integrity Utilities Version 1" on page 30

Migrating from IMS Library Integrity Utilities Version 2 Release 1 (non BPE-based DBRC)

When you migrate Integrity Checker from Version 2 Release 1 to this release of the product and BPE-based DBRC is not used, complete the following migration steps.

About this task

The global option modules, LICON data sets, JCL, and the control statements that are used in IMS Library Integrity Utilities Version 2 Release 1 are compatible.

However, in IMS Library Integrity Utilities Version 2 Release 2, alias name DSPCRTR0 is not assigned for the FABLRT0 module by default. Therefore, you must perform the following steps to customize the LIU load modules to work with DBRC module DSPCRTR0.

Procedure

1. If one of the following conditions is met, shut down the IMS online systems in which Integrity Checker is activated.
   - You plan to install the IMS Library Integrity Utilities load modules (LIU load modules) into the existing IMS Library Integrity Utilities library.
   - You plan to merge the LIU load modules into the IMS SDFSRESL library.

2. Install IMS Library Integrity Utilities.
   - Complete the SMP/E installation of IMS Library Integrity Utilities.

3. Assign alias name DSPCRTR0 for the FABLRT0 module or merge Integrity Checker load modules into the IMS SDFSRESL library.
• If the Integrity Checker load modules were not merged into the IMS SDFSRESL library in the IMS Library Integrity Utilities Version 2 Release 1 environment, assign alias name DSPCRTR0 to the FABLRTR0 load module.

For more information about completing this task, see “Method 1. Customizing LIU load modules by creating alias name DSPCRTR0” on page 60.

• If the Integrity Checker load modules were merged into the IMS SDFSRESL library in the IMS Library Integrity Utilities Version 2 Release 1 environment, merge the Integrity Checker load modules of IMS Library Integrity Utilities Version 2 Release 2 into the IMS SDFSRESL library.

Tip: You do not need to run the FABLUMD1 job, which receives and applies USERMOD to install the FABLRTR0 module into your IMS SDFSRESL library because the FABLRTR0 module is compatible and it does not need to be updated.

For more information about completing this task, see “Method 2. Customizing LIU load modules by merging into the IMS SDFSRESL library” on page 63.

4. Restart the IMS online systems in which you want to activate Integrity Checker.

If you did not shut down the IMS online systems in the preceding step, shut down and restart the IMS online systems.

Migrating from IMS Library Integrity Utilities Version 2 Release 1 (BPE-based DBRC)

When you migrate Integrity Checker from Version 2 Release 1 to this release of the product and BPE-based DBRC is used, complete the following migration steps.

About this task

The global option modules, LICON data sets, JCL, and the control statements that are used in IMS Library Integrity Utilities Version 2 Release 1 are compatible.

However, in IMS Library Integrity Utilities Version 2 Release 2, alias name DSPCRTR0 is not assigned for the FABLRTR0 module by default. Therefore, you must perform the following steps to customize the LIU load modules to work with DBRC module DSPCRTR0.

Procedure

1. Install IMS Library Integrity Utilities.

   Complete the SMP/E installation of IMS Library Integrity Utilities.

2. Assign alias name DSPCRTR0 for the FABLRTR0 module or merge Integrity Checker load modules into the IMS SDFSRESL library.

   • If the Integrity Checker load modules were not merged into the IMS SDFSRESL library in the IMS Library Integrity Utilities Version 2 Release 1 environment, assign alias name DSPCRTR0 to the FABLRTR0 load module.

   For more information about completing this task, see “Method 1. Customizing LIU load modules by creating alias name DSPCRTR0” on page 60.

   • If the Integrity Checker load modules were merged into the IMS SDFSRESL library in the IMS Library Integrity Utilities Version 2 Release 1 environment, merge the Integrity Checker load modules of IMS Library Integrity Utilities Version 2 Release 2 into the IMS SDFSRESL library.
Tip: You do not need to run the FABLUMD1 job, which receives and applies USERMOD to install the FABLRT0 module into your IMS SDFSRESL library because the FABLRT0 module is compatible and it does not need to be updated.

For more information about completing this task, see “Method 2. Customizing LIU load modules by merging into the IMS SDFSRESL library” on page 63.

3. Issue the BPE REFRESH USEREXIT command to reload the Integrity Checker load modules.

For more information about the BPE REFRESH USEREXIT statement, see the topic "BPE REFRESH USEREXIT command" in IMS Commands.

Migrating from IMS Library Integrity Utilities Version 1

When you migrate Integrity Checker from Version 1 to this release of the product, complete the following migration steps.

About this task

The global option modules, JCL, and the control statements that are used in IMS Library Integrity Utilities Version 1 are compatible. However, in IMS Library Integrity Utilities Version 2 Release 2, alias name DSPCRTR0 is not assigned for the FABLRT0 module by default. Therefore, you must customize the LIU load modules to work with DBRC module DSPCRTR0. Also, the LICON data sets that are used in IMS Library Integrity Utilities Version 1 are not compatible. Therefore, you must migrate or re-create the LICON data sets.

Procedure

1. Shut down the IMS online systems in which Integrity Checker is activated.
2. Install IMS Library Integrity Utilities.
   Complete the SMP/E installation of IMS Library Integrity Utilities.
3. Optional: Create global option modules.
   Global option modules that were created by Integrity Checker of IMS Library Integrity Utilities Version 1 can be used in this version of the product. However, Integrity Checker of IMS Library Integrity Utilities Version 2 treats them as the global option modules in which the CHECKRV, the CHKFPSEL, the RECLD, the RECUL, and the RECUPD options are not specified.
   For more information about these options and instructions for changing the global option modules, see the following topics:
   - “JCL requirements for the FABLPGEN program” on page 92
   - "Changing the global option module“ on page 75
4. Migrate or re-create the LICON data sets.
   The format of RDE records that are stored in LICON data sets has been changed in Integrity Checker of IMS Library Integrity Utilities Version 2. Therefore, Integrity Checker cannot process the RDEs that are created by Integrity Checker of IMS Library Integrity Utilities Version 1. You must complete one of the following steps:
   - Migrate the LICON data sets used by IMS Library Integrity Utilities Version 1 to the new format by using the LICON data set migration utility that is provided in IMS Library Integrity Utilities Version 2.
   - Allocate new LICON data sets and create new RDEs.
The following steps explain the procedure to migrate the format of a LICON data set from IMS Library Integrity Utilities Version 1 to the format of IMS Library Integrity Utilities Version 2.

a. Define a backup data set and a new LICON data set with enough free space.

   For more information about allocating LICON data sets, see the following topics:
   - “Estimating the size of the LICON data set” on page 51
   - “Defining and initializing the LICON data set” on page 56

b. Copy the LICON data set to the backup data set.

c. Migrate the original LICON data set from IMS Library Integrity Utilities Version 1 to IMS Library Integrity Utilities Version 2 by using the LICON data set migration utility.

d. Rename the migrated LICON data set name to the original LICON data set name.

   Attention: After you migrate the LICON data set, do not use the LICON data set with Integrity Checker of IMS Library Integrity Utilities Version 1. If the LICON data set is used in Integrity Checker of IMS Library Integrity Utilities Version 1, unexpected results can occur.

   Use the JCL sample shown in Figure 1 on page 32 to migrate the LICON data set. This JCL sample is provided as a member of the SHPSSAMP library.

5. Assign alias name DSPCRTR0 for the FABLRTR0 module or merge Integrity Checker load modules into the IMS SDFSRESL library.

   - If the Integrity Checker load modules were not merged into the IMS SDFSRESL library in the IMS Library Integrity Utilities Version 1 environment, assign alias name DSPCRTR0 to the FABLRTR0 load module.

      For more information about completing this task, see “Method 1. Customizing LIU load modules by creating alias name DSPCRTR0” on page 60.

   - If the Integrity Checker load modules were merged into the IMS SDFSRESL library in the IMS Library Integrity Utilities Version 1 environment, merge the Integrity Checker load modules of IMS Library Integrity Utilities Version 2 Release 2 into the IMS SDFSRESL library.

      Tip: You do not need to run the FABLUMD1 job, which receives and applies USERMOD to install the FABLRTR0 module into your IMS SDFSRESL library because the FABLRTR0 module is compatible and it does not need to be updated.

      For more information about completing this task, see “Method 2. Customizing LIU load modules by merging into the IMS SDFSRESL library” on page 63.

6. Restart the IMS online systems in which you want to activate Integrity Checker.

Example

The following figures present a JCL sample to migrate a LICON data set. This JCL sample is in the SHPSSAMP library, member FABLCNV2.
Figure 1. JCL sample to migrate a LICON data set (Part 1 of 2)
Migrating Advanced ACBGEN

If the load modules are not merged into the IMS SDFSRESL library, no migration tasks are required; you can migrate the utility by replacing and installing the IMS Library Integrity Utilities load module library. Complete this step only if the load modules are merged into the IMS SDFSRESL library.

Procedure

Run the FABQUMD1 job, which is in the SHPSJCL0 JCL library, before merging the load modules of IMS Library Integrity Utilities Version 2 Release 2. This job deletes alias name DFSUACB0 from the IMS LIU SHPSLMD0 library and the LMOD entry of LIU SMP/E CSI of IMS Library Integrity Utilities Version 2 Release 2. You do not need to run the FABQUMD2 job and the FABQUMD3 job. The FABQUMD2 job lists the IMS DFSRRA80 source entry, and the FABQUMD3 job receives and applies USERMOD to modify the IMS DFSRRA80 module so that the module invokes the IMS LIU FABQMAIN module instead of DFSUACB0.

Related tasks:

“Merging Advanced ACBGEN load modules into the IMS SDFSRESL library” on page 352
Chapter 3. Integrity Checker utility

The Integrity Checker utility prevents database corruption caused when IMS applications use incorrect IMS control blocks to get access to a database.

Topics:
- “Integrity Checker overview” on page 36
- “Planning for Integrity Checker configuration” on page 39
- “Activating Integrity Checker” on page 54
- “Maintaining Integrity Checker” on page 67
- “Preventing database corruption with Integrity Checker” on page 79
- “Deactivating Integrity Checker” on page 86
- “Output from Integrity Checker” on page 89
- “Global option module generation macro” on page 91
- “LICON utility reference” on page 98
Integrity Checker overview

Integrity Checker prevents database corruption that is caused by a use of incorrect database descriptions (DBDs).

Subsections:
- “DMB verification function”
- “How the DMB verification function works”
- “Change history report for DBDs and databases” on page 38
- “Supported database organizations” on page 38
- “Program structure” on page 38

DMB verification function

Integrity Checker supports the DMB verification function to prevent database corruptions that are caused by misuse of incorrect DBDLIBs or ACBLIBs.

When IMS databases are accessed by IMS online applications, batch applications, utilities, or IMS Tools jobs, IMS uses the database management block (DMB) to obtain database definition. DMBs are stored in the DBD libraries (DBDLIB) or ACB libraries (ACBLIB) as load modules.

Every IMS job requires a DBDLIB or an ACBLIB. IMS loads the DMB from the DBDLIB or the ACBLIB specified by the job, obtains the DMB, and processes the database based on the information in the DMB. If the DMB is incorrect and IMS uses the incorrect DMB to update the database, the database can become corrupted. For example, database corruptions can occur in the following cases:
- The ACBLIB for a test database is used to update the production database.
- After changing the DBD, the old DBDLIB is used to update the database.

How the DMB verification function works

When an IMS online application, batch application, utility, or IMS Tools job tries to access the database, while the job is being initialized, the Database Recovery Control facility (DBRC) performs authorization processing for the target database.

Integrity Checker stores the correct DBD definition in a control file called the Library Integrity Control data set (LICON data set). During the DBRC authorization process, Integrity Checker compares the DMB that is referred to by IMS and the DMB that is stored in the LICON data set to ensure that IMS is using the correct DMB.

If a mismatch is found between the two DMBs, the DMB verification process modifies the response from DBRC to deny the DBRC authorization request. When the job receives the authorization failure notification, the job terminates without updating the database.

The following figures illustrate how Integrity Checker prevents database corruptions. Figure 3 on page 37 shows DMB verification for IMS online applications, and Figure 4 on page 37 shows DMB verification for batch applications and utilities. The steps that follow the figures correspond to the numbers in the figures.
1. IMS application or utility program uses an incorrect DBDLIB or ACBLIB.

2. The DL/I module requests DBRC authorization.

3. Integrity Checker compares the DMBs, detects a mismatch, and interrupts the DBRC authorization process to deny database authorization.

4. The IMS application or the utility program terminates before it updates the database.

Through these processes, Integrity Checker detects and prevents any access to the database that uses incorrect DMBs.

These figures illustrate the process flows in the non-BPE based DBRC environment. In the BPE-based DBRC environment, Integrity Checker runs as a DBRC Request exit routine to verify the DMBs.

---

**Figure 3. How Integrity Checker prevents database corruption: IMS online applications**

**Figure 4. How Integrity Checker prevents database corruption: batch applications and utilities**
Change history report for DBDs and databases

Integrity Checker keeps historical records of DBDs that were used in DMB verifications. You can print these records to review the change histories and to compare the current version of the DBD with an earlier version.

You can also record the time stamp of the last database access by using the database access recording option. If you activate this option, the time stamp of the last access made by IMS programs that have an update intent, load utility programs, and unload utility programs is recorded. Such records can also be printed in a report, and you can use the records for an audit evidence regarding database access.

Supported database organizations

Integrity Checker supports the following types of database organization:

- HSAM, HISAM, SHISAM, HIDAM, HDAM, INDEX
- PHDAM, PHIDAM, PSINDEX
- DEDB

MSDB and GSAM databases are not supported.

Program structure

Integrity Checker contains the following programs:

- DMB verification modules that do DMB verification
- The utility module that creates, maintains, and prints LICON data sets

Integrity Checker also provides a cataloged procedure and a macro for creating global option modules.
Planning for Integrity Checker configuration

Before you activate the DMB verification function of Integrity Checker, you must understand the resources used by Integrity Checker and design the most suitable Integrity Checker configuration for your IMS environment.

After you design your Integrity Checker configuration, activate Integrity Checker by completing the instructions in “Activating Integrity Checker” on page 54.

Use the following topics to design the Integrity Checker configuration:

- “LIU load module library customization”
- “LICON data sets and global option modules” on page 40
- “Integrity Checker configuration requirements” on page 41
- “Runtime options and environments” on page 47
- “Historical data maintained in LICON data sets” on page 48
- “Considerations for activating Integrity Checker” on page 50

LIU load module library customization

The IMS Library Integrity Utilities load module library (also referred to as LIU load module library) must be customized to use Integrity Checker.

The DMB verification function of Integrity Checker is triggered by IMS jobs (IMS online, application, IMS standard utility, and IMS Tools utility jobs) that access databases. Therefore, unlike other IMS Tools programs that run with JCL in which the EXEC PGM= parameter specifies the utility program name to execute, Integrity Checker must be customized to automatically start DMB verification when a database is accessed.

The DMB verification process runs as part of the DBRC authorization process. Therefore, instead of coding JCL statements, you must customize the LIU load module data set (SHPSLMD0 data set) to work with DBRC module DSPCRTR0.

Important: In IMS Library Integrity Utilities Version 2 Release 1 and earlier, alias name DSPCRTR0 was assigned for the FABLRTR0 module by default, but in IMS Library Integrity Utilities Version 2 Release 2 and later, the alias name is not assigned. You must customize the LIU load module library.

Methods for customizing the LIU load module library

Integrity Checker supports two methods for customizing the LIU load module data set. Compare the two methods and determine the best method for your environment.

Method 1: Defining alias name DSPCRTR0 for FABLRTR0 (recommended method)

Module FABLRTR0 is the program load module that is stored in the LIU load module library. Module DSPCRTR0 is the DBRC router module, which is one of the DBRC authorization modules.

If alias name DSPCRTR0 is defined for the FABLRTR0 module, and the LIU load module library precedes the IMS resident library (SDFSRESL data set) in the STEPLIB DD in JCL and procedures of IMS jobs, when a database is accessed by such jobs, instead of the DBRC module DSPCRTR0, the FABLRTR0 module starts and activates DMB verification. To apply this method, the LIU load module library must be APF-authorized.
• Advantage: You can use the standard SMP/E methods to apply program temporary fixes (PTFs) for both IMS Library Integrity Utilities and IMS.
• Disadvantage: You must modify the STEPLIB DD statement in JCL and procedures of all IMS jobs that access databases.

Method 2: Merging Integrity Checker load modules into the IMS resident library (SDFSRESL)

If all the Integrity Checker modules (FABL* members) in the LIU load module library are merged into the IMS SDFSRESL data set, when a database is accessed, the FABLRTR0 module, which is link-edited to the DBRC module DSPCRTR0 module, starts and activates DMB verification. Before the modules are merged, the FABLRTR0 module and DBRC module DSPCRTR0 must be link-edited to create one DSPCRTR0 module, and the module entry point must be set to FABLRTR0. Before the modules are merged, a backup of the IMS SDFSRESL data set must be created.

• Advantage: You do not need to modify the STEPLIB DD statement in each JCL and procedure of IMS jobs that access databases.
• Disadvantage: When you apply PTFs for IMS Library Integrity Utilities and IMS, the following additional steps are required:
  – For IMS Library Integrity Utilities, after you apply a PTF, you must merge the members again.
  – For IMS, if the PTF updates the DSPCRTR0 module, you must restore the DSPCRTR0 module before you apply the PTF. After you apply the PTF, create a backup of the SDFSRESL data set, and then link-edit FABLRTR0 and DSPCRTR0 again.

Integrity Checker supports multiple versions of IMS with one module. Therefore, with either method, if the version of IMS is supported by IMS Library Integrity Utilities, the same LIU load module data set can be used.

**LICON data sets and global option modules**

LICON data sets and global option modules are unique resources that are required by Integrity Checker.

**LICON data sets**

*LII Integrity control data sets* (LICON data sets) are KSDS data sets that are the repositories for registered DMB entries (RDEs). RDEs contain database management block (DMB) information that is used in DMB verification. In the LICON data sets, RDEs are stored as KSDS records. At least one LICON data set must be defined and initialized.

Integrity Checker obtains the DMB information that IMS used to load the database, stores the information in the RDE as the correct DMB information, and refers to that DMB information to verify the DMB information obtained from the DBDLIB or ACBLIB that is referenced by IMS jobs.

In addition to the DMB information, RDEs contain the following information:
• Runtime options that are applied to DMB verification.
• Time stamp of the last database access made by load utility programs, unload utility programs, and utilities or application programs with an update intent in the PSB.
When a DBD is updated, Integrity Checker updates the corresponding RDE accordingly.

In addition to the RDE that contains the latest DMB information, Integrity Checker also maintains historical copies of RDEs. You can use those copies to review the DBD information before a DBD change, or when you restore DBDs, you can use the copies to restore RDEs.

**Global option modules**

Global option modules are the configuration definition modules of Integrity Checker. Each global option module defines the LICON data set and the runtime options that are applied to DMB verification. IMS Library Integrity Checker does not provide global option modules, so you must create at least one global option module.

IMS Library Integrity Utilities provides the FABLPGEN program for creating global option modules. The LICON data set name and runtime options are defined through FABLPGIN macro control statements of the FABLPGEN program.

After running the FABLPGEN program, the source code must be assembled and link-edited to a load module. The created global option module must be stored in the LIU load module library.

**Integrity Checker configuration requirements**

Because DMB verification works within DBRC authorization processing, DBRC must be active in your IMS environment. The number of LICON data sets and global option modules in an Integrity Checker configuration depends on how the DBRC environment is configured.

The basic rules for designing an Integrity Checker configuration are as follows:

- One LICON data set must be created for each set of RECON data sets. (A set consists of RECON1, RECON2, and RECON3.)
- Global option modules contain the names of the LICON data sets. Therefore, if your environment requires multiple LICON data sets, you must prepare the same number of global option modules.
- Global option modules must be named using the following naming convention:
  - **LIU@INST**
    
    The default name for the global option module. If your environment requires only one LICON data set, use this name.
  
  - **LIU@imsid**
    
    If your environment requires multiple LICON data sets, use this format. For *imsid*, use the IMS ID of each IMS subsystem.

    When multiple global option modules exist, Integrity Checker determines the global option module to use from the IMS ID. If no corresponding global option module is found for an IMS ID, Integrity Checker uses the default module LIU@INST. Even if the corresponding global option module is found, if some options are not defined in that module, the options that are defined in LIU@INST are applied. If LIU@INST is not found or some options are not defined in the LIU@INST, the system default values are used.
**Tip:** At sign (@) is a code-page-dependent character. If you are working in an environment where you cannot use the at sign (@), name the modules LIUGINST or LIUGimsid.

Use the following examples to design an Integrity Checker configuration for your environment.

- **“Single IMS subsystem configuration example”**
- **“Multiple IMS subsystems configuration example: Data-sharing environment”** on page 43
- **“Multiple IMS subsystems configuration example: Non-data-sharing environment”** on page 44
- **“Multiple IMS subsystems configuration example: Multiple data-sharing environments”** on page 45
- **“Multiple IMS subsystems configuration example: XRF complex”** on page 46

As a rule, the same runtime options must be defined for each LICON data set. To simplify the explanations, these examples are cases where one global option module refers to one LICON data set.

**Tip:** Certain runtime options can have different values within one LICON data set. To define different runtime option values, instead of assigning alias names for global option modules as shown in the examples, create one global option module for each IMS ID. For more information about the runtime options, see **“Global option module generation macro”** on page 91.

For IMS environments that are not explained in these examples, such as IMSplex, the basic rules are the same. For any IMS environment, you can follow these basic rules to design Integrity Checker configuration:

- One LICON data set for one set of RECON data sets
- One global option module for one LICON data set

**Single IMS subsystem configuration example**

The following figure shows the simplest configuration.

In this environment, one IMS subsystem is used and one set of RECON data sets is used, so the Integrity Checker resources that are required in this environment are as follows:

- Number of LICON data sets: 1
- Number of global option modules: 1

Because only one global option module is required, the name of the global option module is LIU@INST.
Multiple IMS subsystems configuration example: Data-sharing environment

The following figure illustrates a configuration where two IMS subsystems share databases.

In this environment, two IMS subsystems are used and one set of RECON data sets is used, so the Integrity Checker resources that are required in this environment are as follows:

- Number of LICON data sets: 1
- Number of global option modules: 1

Because only one global option module is required, the name of the global option module is LIU@INST.

Figure 5. Integrity Checker configuration for a single IMS subsystem

Multiple IMS subsystems configuration example: Data-sharing environment
Multiple IMS subsystems configuration example: Non-data-sharing environment

The following figure illustrates a configuration where two IMS subsystems use a different set of databases. In such an environment, a LICON data set must be created for each IMS subsystem. Each LICON data set requires one global option module.

In this environment, two IMS subsystems are used and two sets of RECON data sets are used, so the Integrity Checker resources that are required in this environment are as follows:

- Number of LICON data sets: 2
- Number of global option modules: 2

Because two global option modules are required, the names of the global option modules are LIU@IMSA for IMS subsystem IMSA, and LIU@IMST for IMS subsystem IMST.

Figure 6. Integrity Checker configuration in a data-sharing environment
Multiple IMS subsystems configuration example: Multiple data-sharing environments

The following figure illustrates a configuration for multiple data-sharing environments. In each data-sharing environment, two IMS subsystems use the same set of databases. In such environments, a LICON data set must be created for each data-sharing environment.

Generally, when four IMS subsystems exist, four global option modules are required. However, as shown in the figure, if you create a global option module for each LICON data set and define an alias name for each global option module, you can design the Integrity checker configuration with two global option modules.

Create global option module LIU@IMSA for IMS subsystem IMSA, and then define alias name LIU@IMSB for LIU@IMSA so that IMS subsystem IMSB can also use LIU@IMSA. Do the same for IMS subsystems IMSC and IMSD.

**Tip:** If you create two global option modules (LIU@IMSA and LIU@IMSB), the LICON data set names and the runtime options must be the same between the two global option modules. By assigning an alias name and sharing a global option module, the same values are automatically applied; you can prevent errors that might be caused by inconsistency.

Figure 7. Integrity Checker configuration in a non-data-sharing environment

Chapter 3. Integrity Checker utility 45
Multiple IMS subsystems configuration example: XRF complex

To use Integrity Checker in an XRF complex, the active IMS subsystems and the alternate IMS subsystems must use the same LICON data set. If they use different LICON data sets, the change log of the DMB is not inherited during takeover, and Integrity Checker might not detect the DMB inconsistency or might deny DBRC authorization even though the correct DBDLIB or ACBLIB is used.

If the number of active IMS subsystems is one, the number of required LICON data set is also one. The name of the global option module is LIU@INST.

If multiple active IMS subsystems exist, define a LICON data set for each set of RECON data sets, and create one global option module for each LICON data set.

For example, in an XRF complex shown in the following figure, create a global option module for each active IMS subsystem. Create LIU@IMS1 and LIU@IMS3, and then assign alias name LIU@IMS4 for LIU@IMS3.

Then, to apply the same runtime options to the active IMS subsystems and the alternate IMS subsystems, assign alias to the global option modules so that the alternate IMS subsystems can also use the same global option modules.

Specifically, IMS subsystems IMS1 and IMS2 are in a same XRF complex, so the two subsystems must use the same LICON data set. Therefore, assign alias name LIU@IMS2 for LIU@IMS1. IMS subsystems IMS5 and IMS6 are alternate IMS subsystems for IMS3 and IMS4, so all these subsystems must use the same LICON data set. Therefore, assign alias names LIU@IMS5 and LIU@IMS6 for LIU@IMS3.
Runtime options and environments

RDEs contain runtime options for DMB verification. Some runtime options can be defined differently for each IMS processing environment.

How the runtime options are specified

The runtime options are commonly used across the environment that uses the global option module. The global default values that are applied to the options when creating new RDEs can be specified for each database.

The runtime options are defined in the global option module. You can specify the following runtime options:

- The data set name of the LICON data set
- DMB verification method (single-step or double-step)
- Routing codes and descriptor codes of Integrity Checker messages
- Option values to control the processing of Integrity Checker
- Global default values for the options used in creating new RDEs:
  - DMB verification option
  - Database access record option
  - Number of expired RDEs to maintain

Figure 9. Integrity Checker configuration in an XRF complex
You specify the option values in the global option module by using the global option module generation macro. The option values that you specify are copied to the RDE when an RDE is created.

Generally, you can use the option values without changing them, but by using the LICON utility, you can change the option values in RDEs after they are created. By changing the values, you can apply different options for each database. For more information about these options, see “Global option module generation macro” on page 91.

How the runtime options are applied

Some runtime options can be set differently for each environment that Integrity Checker operates. Such options are referred to as DMB verification options.

DMB verification options control the processing and the action taken when a DMB mismatch is found. You can specify the options for each of the following IMS processing environments:

- Online IMS subsystem
- Batch program
- User load program
- Batch image copy utility
- Database recovery utility

For each environment, you can specify:

- Whether to activate DMB verification.
- Whether DMB verification sets a nonzero return code for the database authorization request to deny it, or only issues a warning message and registers the DMB to the LICON data set.

Using these options, you can control the behavior of DMB verification differently in each environment.

Recommendation: In all environments, set the option to deny the database authorization request when a DMB mismatch is found. By setting this option, you can prevent database corruption when an incorrect DBDLIB or ACBLIB is used.

Attention: Use the option Only issue a warning message and register the DMB to the LICON data set with caution. Use this option only when you intend to change the DBD, and you want the updated DBD reflected in the RDE so that the new DMB information is used in future DMB verifications. When this option is used, database corruption caused by use of an incorrect DBDLIB or ACBLIB cannot be prevented.

For example, if you want to use the DBD that is specified by a user load program for future DMB verifications, but you want to restrict DBD changes in other environments, you can have Integrity Checker issue a warning message and update the RDE in the user load program jobs, but deny database authorization when a DMB mismatch is found in other environments.

Historical data maintained in LICON data sets

Integrity Checker stores historical copies of RDEs in the LICON data sets. RDEs record database access logs for database update, load, and unload applications.
Historical copies of DBD definition

Integrity Checker maintains historical copies of RDEs. Each of these copies contains a part of the database description (DBD) that is used in DMB verifications. Whenever a DBD is changed, Integrity Checker creates a new RDE that contains the latest DMB information. The historical copies of RDEs can be used to track the changes made to DBDs.

The content of the current RDE and historical copies of RDEs can be printed in reports. You can use the reports to examine the information in DBDs, review the changes made to DBDs, or to compare the difference between the current DBD and the DBDs used in the past.

To print the content of an RDE in a report, use the LIST.DB command of the LICON utility. For more information, see the following topics:

- To print a report, see "LIST.DB command" on page 113.
- For report field descriptions, see "Output from Integrity Checker" on page 89.

Database access recording option

Integrity Checker records the time stamp of the last database access that is made by update, load, and unload applications in the RDE together with the IMS subsystem name. Such information can be printed in a report, which can be used as evidence in database auditing.

To record database accesses, make the specifications in the RDE by using the global option module or the LICON utility. To use the database access recording option, specify the option for each of the following database access types:

<table>
<thead>
<tr>
<th>Database access</th>
<th>Access type identified by Integrity Checker</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load utility</td>
<td>Load access</td>
</tr>
<tr>
<td>Reorganization utility</td>
<td>Load access</td>
</tr>
<tr>
<td>Unload utility</td>
<td>Unload access</td>
</tr>
<tr>
<td>Recovery utility</td>
<td>Update access</td>
</tr>
<tr>
<td>Utility or application program with the PCB processing option (PROCOPT) of A, I, R, or D</td>
<td>Update access</td>
</tr>
</tbody>
</table>

For instructions to specify these options, see the following topics:

- To set the options in the global option module, see "Global option module generation macro" on page 91.
- To update the options in RDEs, see "INIT.DB command" on page 101.

When Integrity Checker is activated in an IMS online environment, Integrity Checker records access information only for the first database authorization request, and the recorded time is not updated while the IMS online environment is active. To have the recorded information updated while the IMS online environment is active, the ACCESS parameter of the DATABASE macro statement must specify UP or EX (during the system definition stage), or the parameter for the /START DB ACCESS= command must specify UP or EX.
Restrictions: The database access recording option can be used for the utilities or the application programs that are supported by Integrity Checker. However, the following restrictions apply:

- This option is not effective for online reorganization functions or online reorganization utilities that are provided by IMS or IMS Tools.
- For a database access through the High-Speed DEDB Direct Reorganization utility, Integrity Checker identifies the database access type as a database update.
- Database access for load or unload operations is recorded only when the operations are done by IMS standard utilities or IMS Tools utilities.
- If a utility or an application program ends with an error after the Integrity Checker DMB verification process ends successfully, the database access information that is recorded in the RDE is updated in the same way as when the utility or the application program ended successfully, even though the utility or application program ended with an error.
- For index databases or secondary index databases that have no PCBs, Integrity Checker does not record database access information in the RDEs except when the database operation is done by IMS Database Reorganization Expert.

Considerations for activating Integrity Checker

Before you activate Integrity Checker, review these considerations.

Cases where DMB verification is not done
DMB verification is not done in certain environments or for certain application jobs.

For more information, see “Restrictions: Cases where DMB verification is not done” on page 79.

Size of the LICON data sets
Before creating LICON data sets, you can estimate the required storage for the LICON data sets.

For more information, see “Estimating the size of the LICON data set” on page 51.

LICON data set serialization consideration
If you want more than one MVS™ system to access the LICON data set, you must serialize the LICON data set. To access the LICON data set from more than one MVS system, Global Resource Serialization (GRS) or a similar global enqueue product must be installed.

For more information, see “Serializing the LICON data set” on page 57.

RACF security considerations
If you want to protect the LICON data sets with RACF or if you plan to use Integrity Checker in IMS Database Recovery Facility jobs, you must modify RACF security.

For more information, see “Setting up RACF security” on page 58.

Considerations when you alter the definition of an online HALDB or an online DEDB
When you alter the definition of an online HALDB database or an online DEDB database with the following IMS command or IMS utility, you must temporarily stop the DMB verification process.

- Use the INITIATE OLREORG command (with the ALTER option) and the online change (OLC) function to alter the definition of an online HALDB database.
Use the DEDB Alter utility (DBFUDA00) to alter the definition of an online DEDB database.

After you alter the definition, restart the DMB verification process with new RDEs. Without these steps, the DMB verification process uses the old definitions to verify the DMBs. Therefore, the DMB verification process might deny database authorization requests that use correct IMS control blocks.

For detailed instructions, see the following topics:
- “Altering the definition of a DMB verification-enabled online HALDB by using the HALDB alter function” on page 72
- “Altering the definition of a DMB verification-enabled online DEDB by using the DEDB Alter utility (DBFUDA00)” on page 73

Consideration when you change the maximum size of OSAM data sets for a HALDB

When you change the maximum size of OSAM data sets for a HALDB from 4 GB to 8 GB or 8 GB to 4 GB, you must use the LICON utility and manually create RDEs to reflect the change for the DMB verification process.

For more information, see “Changing the maximum OSAM data set size for a DMB verification-enabled HALDB” on page 75.

Estimating the size of the LICON data set

Before you create LICON data sets, you can estimate the required storage for the LICON data sets.

About this task

Two methods are available for estimating the LICON data set size. The first method is more complicated but results in a precise data set size. The other method is simpler, but results in only an approximate data set size. You can use the simpler estimation method if the number of database segments is less than 20.

Tips:
- To accommodate any future increase in the number of databases, partitions, or DEDB areas, make the LICON data set size larger than the value that you calculate. For example, when you allocate a LICON data set, increase the calculated primary allocation size by 10% to 20%, and increase the secondary allocation size by approximately 10% of the primary allocation size.
- When you create multiple LICON data sets, you can create them on the same volume or on different volumes.

Procedure

For each LICON data set, use one of the following methods to estimate the data set size:

Estimating the precise LICON data set size

Use the following formula to calculate the precise LICON data set size:
LICON data set size = total_size_required_for_full-function_databases
+ total_size_required_for_HALDBs
+ total_size_required_for_DEDBs

**total_size_required_for_full-function_databases**
The total size of all the full-function databases that are to be processed.

For each full-function database, calculate the required size by using the following formula:

\[(450 + 32 \times \text{number of DSGs} + 52 \times \text{number of segments} + 180 \times \text{access_info_size}) \times 16\]

**Note:** DSG stands for data set group.

**total_size_required_for_HALDBs**
The total size of all the HALDB partitions that are to be processed.

For each HALDB partition, calculate the required size by using the following formula:

\[(450 + 32 \times \text{number of DSGs} + 52 \times \text{number of segments} + 180 \times \text{access_info_size}) \times 16\]

**total_size_required_for_DEDBs**
The total size of all the DEDB areas that are to be processed.

For each DEDB area, calculate the required size by using the following formula:

\[(220 + 48 \times \text{number of segments} + 180 \times \text{access_info_size}) \times 16\]

**Notes:**
- If you do not enable the database access recording option (activated by the RECUPD, RECLD, or RECUL keyword in the global option module), access_info_size is 0. When this option is enabled, access_info_size increases by 1 for each keyword that you specify. For example, if you enable the RECUPD option (one keyword), access_info_size is 1. If you enable RECUPD and RECLD options (two keywords), access_info_size is 2.

**Related reading:** For instructions to activate this option, see “Global option module generation macro” on page 91.
- For these formulas, the units are in bytes.
- Each formula is multiplied by 16 for storing histories of database definitions.

**Estimating the approximate LICON data set size**

Use the following formula to calculate the approximate LICON data set size:

LICON data set size = (32 \times \text{number of full-function databases} 
+ 32 \times \text{number of HALDB partitions} 
+ 16 \times \text{number of DEDB areas}) \text{ KB}
number_of_full-function_databases
The number of full-function databases (excluding HALDBs) that are to be processed.

number_of_HALDB_partitions
The number of HALDB partitions that are to be processed.

number_of_DEDB_areas
The number of DEDB areas that are to be processed.
Activating Integrity Checker

Activating Integrity Checker involves preparing the global option modules, LICON data sets, and LIU load modules, and other steps.

Before you begin

Before you activate Integrity Checker, plan an Integrity Checker configuration for your environment, as described in “Planning for Integrity Checker configuration” on page 39.

About this task

To activate Integrity Checker, you first prepare the global option modules and the LICON data sets. Then, by using the DBD or ACB library, register the DMB information for your databases in the LICON data set. When this registration is done, customize the LIU load modules by using either of the following methods:

• Create an alias of the DSPCRTR0 module and add DD statements to JCL or procedures of DBRC, IMS batch application, IMS utility, and IMS Tools (recommended method).
• Merge load modules into the IMS SDFSRESL library.

Finally, activate Integrity Checker by restarting IMS online and running IMS batch applications, IMS utilities, and IMS Tools jobs.

In both IMS batch and online environments, the steps to activate Integrity Checker are the same. These steps can be applied when you reactivate Integrity Checker after Integrity Checker is deactivated.

Procedure

To activate Integrity Checker, complete the following steps:

1. “Setting up the global option modules”
2. “Setting up the LICON data sets” on page 55
3. “Setting up RACF security” on page 58
4. “Customizing LIU load modules” on page 59
5. “Configuring for a BPE-based DBRC environment” on page 65
6. “Verifying that Integrity Checker is activated” on page 65
7. “Restarting IMS online and running IMS batch application, IMS utility, and IMS Tools jobs” on page 66

Setting up the global option modules

Create global option modules and, optionally, assign alias names to the global option modules. At least one global option module must be present when Integrity Checker is started because Integrity Checker uses it to obtain the name of the LICON data set that it uses.

Before you begin

See “Integrity Checker configuration requirements” on page 41 and determine the number of global option modules to create. Also, determine whether to share a global option module across multiple IMS subsystems.
Procedure

1. Create global option modules.

To create global option modules, use the FABLPGEN procedure and provide the control statements by using the SYSIN input stream. The FABLPGEN procedure is in the SHPSSAMP data set. For an instruction, see “Global option module generation macro” on page 91.

Tip: To avoid activating Integrity Checker until all of your installation activities are complete, have no global option modules created.

Related reading: "Options applied to RDEs when multiple global option modules exist with different effective ranges" on page 76

2. If you want more than one IMS subsystems to use a set of options that are defined in a single global option module, use the linkage editor to assign an alias name to the global option module.

In a database sharing environment where multiple IMS subsystems share databases, the LICON data set and option values that are defined in the global option module must be the same across the IMS subsystems. Assigning aliases is beneficial in such an environment as well as in XRF environments.

Use the following job to assign an alias name to the global option module.

```//LKED JOB //L EXEC PGM=IEWL,PARM='XREF,LIST' //SYSPRINT DD SYSOUT=* //SYSPRINT DD SYSOUT=* //SYSLMOD DD UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)), // SPACE=(1024,(100,10),RLSE),DISP=(,DELETE) //SYSLMOD DD DISP=SHR,DSN=HPS.SHPSLM0 //SYSLIN DD * INCLUDE SYSLMOD(LIU@IMSA) /* Global option module for IMSID=IMSA */ ALIAS LIU@IMSB /* Alias for IMSID=IMSB */ NAME LIU@IMSA /* Original name for IMSID=IMSA */ /*```

Setting up the LICON data sets

Define and initialize LICON data sets, create RDEs to register the correct DMB information, and serialize the LICON data sets. At least one LICON data set must be present to activate Integrity Checker.

Before you begin

See “Integrity Checker configuration requirements” on page 41 to determine the number of LICON data sets to create.

Procedure

The following steps describe how to set up a single LICON data set. If more than one LICON data set is required for your environment, complete the following steps for each LICON data set.

1. “Defining and initializing the LICON data set” on page 56
2. “Creating an RDE to register DMB information” on page 56
3. “Serializing the LICON data set” on page 57
Defining and initializing the LICON data set

Define the LICON data set by using the DEFINE cluster command, and initialize it by using the INIT.LICON command of the LICON utility.

Procedure

1. Define the LICON data set by using the DEFINE CLUSTER command.

   Sample JCL is in the SHPSJCL0 library, member FABLINIT. The following figure shows the DEFLICON step of the sample JCL, which defines the LICON data set.

   Specify the values for the following parameters. For the CYL parameter, specify the size that you calculated in "Estimating the size of the LICON data set" on page 51.

```
//DEFLICON EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*                  
//SYSIN DD *                              
DELETE (imshlq.licondsn) CLUSTER ERASE PURGE
SET MAXCC=0
DEFINE CLUSTER (NAME(imshlq.licondsn) - 
   INDEXED -                 
   KEY (44 0) -              
   SHR(3 3) -                
   NOREUSE -                 
   VOL (liconvol) -          
   CYL (pri sec) -           
   RECSZ (4096 32760) -      
   FREESPACE (xx xx)) -      
   DATA(NAME(imshlq.licondsn.DATA)) - 
   INDEX(NAME(imshlq.licondsn.INDEX))
```

2. Initialize the LICON data set by using the INIT.LICON command of the LICON utility.

   Sample JCL is in the SHPSJCL0 library, member FABLINIT. The following figure shows the INILICON step of the sample JCL, which initializes the LICON data set.

   Add the load module data sets that contain the global option module and the LIU load module library to the STEPLIB concatenation.

```
//INILICON EXEC PGM=FABLIU00
//STEPLIB DD DISP=SHR,DSN=HPS.SHPSLMD0 
//     DD DISP=SHR,DSN=IMSVS.SDFSRESL 
//FABLPRINT DD SYSOUT=*               
//FABLIN DD *                         
INIT.LICON                             
```

Creating an RDE to register DMB information

Create an RDE to register the correct DMB information by using the LICON utility.

About this task

This task is optional because if you do not create an RDE with the LICON utility, Integrity Checker creates an RDE automatically when the database is accessed for
the first time after Integrity Checker is activated. In this case, Integrity Checker creates an RDE for each database that is accessed while IMS is online. Therefore, the performance of IMS online processing might decline depending on the number of databases. Consider creating RDEs manually by using the LICON utility to avoid performance degradation.

If you want Integrity Checker to create RDEs automatically, you can skip this task.

**Procedure**

Use the LICON utility to create an RDE.

Ensure that you provide the following information with the JCL statements:

- Specify the DBD library or the ACB library that contains the DMB information to be used in DMB verifications.
- To verify the user exit routine, add the data set that contains the user exit routine to be used in DMB verifications to the STEPLIB concatenation.

Provide the INIT.DB command in the FABLIN input stream. The RDE is created in the LICON data set that is specified by the global option module found in the STEPLIB concatenation.

The following JCL example is for the LICON utility. INIT.DB DBD(*) specifies that an RDE is created for every DBD member in the DBD library IMSVS.DBDLIB.

```
//LICJOB   JOB
//            EXEC PGM=FABL1U00
//STEPLIB   DD DISP=SHR,DSN=HPS.SHPSLMD0
//            DD DISP=SHR,DSN=IMSVS.SDFSRESL
//DBDLIB   DD DISP=SHR,DSN=IMSVS.DBDLIB
//FABLPRNT  DD SYSOUT=*     
//FABLIN  DD *
//INIT.DB DBD(*)
/*
```

**Serializing the LICON data set**

This step is optional. If you want more than one MVS system to access the LICON data set, you must serialize the LICON data set.

**Procedure**

To access the LICON data set from more than one MVS system, you must install Global Resource Serialization (GRS) or a similar global enqueue product. GRS processes the resource as a global resource. The global enqueue product must propagate the enqueues to all MVS hosts. If the enqueues are not propagated to all hosts that have access to the LICON data set, the data set becomes corrupted and unusable.

The LICON data set is enqueued with the following parameters at the SYSTEMS level:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>QNAME</td>
<td>'FABLICON'</td>
</tr>
<tr>
<td>RNAME</td>
<td>The name of the LICON data set</td>
</tr>
</tbody>
</table>
Setting up RACF security

You can optionally set up RACF security to protect LICON data sets. If you plan to activate Integrity Checker in IMS Database Recovery Facility jobs and if DBRC command and API request authorization support is enabled with RACF, ensure that appropriate permission is given to users.

Procedure

Complete the following steps to set up RACF security:

- "Setting up security for LICON data sets"
- "Setting up security for IMS Database Recovery Facility jobs" on page 59

Setting up security for LICON data sets

This step is optional. If you want to protect the LICON data sets with RACF, complete this step.

About this task

The following procedure provides a brief overview of the security setting for LICON data sets.

For more information about the security setting of the started tasks, see the topics that describe how to associate started procedures and jobs with user IDs in the z/OS Security Server RACF System Programmers Guide and in the z/OS Security Server RACF Security Administrators Guide.

Procedure

To use RACF to protect the LICON data sets, define data set profiles for the LICON data sets. Also, because LICON data sets are accessed in each environment, consider the following requirements in both IMS online environment and batch environment.

IMS online environment

For IMS online processing, access to the LICON data set is done by the DBRC region. In this case, the DBRC region must be assigned a user ID, preferably by using the RACF STARTED class. This assignment can also be done by using the RACF started task table (ICHRIN03) or the USER= and PASSWORD= values on the DBRC job or the task JCL. Give UPDATE access authority to the LICON data set for the user ID assigned to the DBRC region.

The following list contains an example of the statements that are used to create and enable LICON data set protection by using RACF.

```
AG licongrp
AU liconusr DFLTGRP(licongrp)
RDEF STARTED dbrcrgn.* STDATA(USER(liconusr) GROUP(licongrp))
ADDSD liuhlq.licon UACC(NONE)
PE liuhlq.licon ID(licongrp) ACCESS(UPDATE)
```

```
AG licongrp
  Creates a RACF group named licongrp.
```
AU liconusr DFLTGRP(licongrp)
   Creates a RACF user ID of liconusr and assigns the default group licongrp.

RDEF STARTED dbrcrgn.* STDATA( USER(liconusr) GROUP(licongrp))
   Defines the STARTED class profile, which will assign the user ID to the dbrcrgn procedure. This statement assumes that the PDS member name of the procedure that is started is dbrcrgn. This statement assigns user ID liconusr to the started procedure.

ADDS D liuhlq.licon UACC(NONE)
   Defines a data set profile for the LICON data set.

PE liuhlq.licon ID(licongrp) ACCESS(UPDATE)
   Gives the licongrp group UPDATE access to the LICON data set.

**Batch environment**

The user ID created for the DBRC online region cannot be used because jobs can be submitted by many different users. The user ID will be used if the ID is correctly assigned through the use of the RACF STARTED class by coding entries for job names in the STARTED class and associating those entries with the user ID. However, the STARTED class is used only when the batch jobs are started with the MVS START command. Jobs submitted to a JES reader will not invoke a call to the STARTED class. Therefore, they might not be assigned a user ID that allows appropriate access to the LICON data set. In this case, all users authorized to run batch jobs must be identified and connected to a group that has UPDATE access to the LICON data set.

**Setting up security for IMS Database Recovery Facility jobs**

If you plan to activate Integrity Checker in IMS Database Recovery Facility jobs, complete this step.

**Procedure**

When you activate Integrity Checker in an IMS Database Recovery Facility job, Integrity Checker uses the DBRC command utility and the DBRC API. If you protect the DBRC command and DBRC API request authorization support with RACF by permitting appropriate user access to the profiles, you must permit appropriate users of jobs to use the following DBRC commands and DBRC API requests:

**DBRC command**

LIST.RECON

**DBRC API requests**

- STARTDBRC
- STOPDBRC
- RELBUF
- QUERY,TYPE=DB
- QUERY,TYPE=PART

For more information about the RACF settings for DBRC commands and DBRC API requests, see the topic "DBRC security" in *IMS System Administration*.

**Customizing LIU load modules**

Customize the LIU load modules to activate Integrity Checker.
Before you begin

See “LIU load module library customization” on page 39 to determine which method you use to customize the LIU load modules.

Procedure

Customize the LIU load modules by using either of the following methods:
- “Method 1. Customizing LIU load modules by creating alias name DSPCRT0”
- “Method 2. Customizing LIU load modules by merging into the IMS SDFSRESL library” on page 63

Method 1. Customizing LIU load modules by creating alias name DSPCRT0

Create an alias name DSPCRT0, APF-authorize the LIU load module library, and add DD statements to JCL and procedures of DBRC, IMS batch applications, IMS utilities, and IMS tools.

Procedure

1. Create alias name DSPCRT0 by link-editing the FABLRT0 load module. Complete this step if either of the following conditions apply:
   - You are activating Integrity Checker for the first time.
   - You removed the alias name DSPCRT0 by completing the steps in “Deactivating Integrity Checker when IMS Library Integrity Utilities is installed as a component of an IMS tools solution pack” on page 87 and you want to reactivate Integrity Checker. If you are reactivating Integrity Checker and the DSPCRT0 alias that was created in the last activation still exists, you can skip this step. You can use the following JCL example to create alias name DSPCRT0. This JCL is in the SHPSJCL0 library, member FABLALSC. When the job ends, confirm that the return code is 0.

```
//FABLALSC
//**--------------------------------------------------------------------
//**     STEP1: Add the alias DSPCRT0
//**--------------------------------------------------------------------
//LINK EXEC PGM=IEWL,REGION=0M,
//  PARM='SIZE=(880K,64K),LET,LIST,NCAL,RENT,REFR,XREF'
//  IEWL = IEWBLINK
// SYSIN DD SYSIN=*
//SYSPRINT DD SYSOUT=*  //CAUTION!!!
//SYSLMOD DD DISP=OLD,DSN=LIU.SHPSLMD0 LIU target load module lib
//SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(10,1))
//  CAUTION!!!
//  //** SPECIFY SHPSLMD0 TO THE INCLUDE STATEMENT.
//  //** IF YOU SPECIFY AHPSMDO TO THE INCLUDE STATEMENT,
//  //** THE NON-ACCEPTED CHANGES WILL BE DELETED.
//SYSIN DD *
//  ENTRY FABLRT0
//  INCLUDE SYSLMOD(FABLRT0)
//  ALIAS DSPCRT0
//  NAME FABLRT0(R)
//**--------------------------------------------------------------------
```
2. APF-authorize the LIU load module library.
   The LIU load module library and the load module data sets that contain the
   global option modules must be APF-authorized.

   Use the z/OS SETPROG commands to APF-authorize the load libraries:
   SETPROG APF,ADD,DSNAME=LIU.SHPSLMDO,VOLUME=volume

   For more information about the SETPROG command, see z/OS MVS System
   Commands.

3. Add DD statements to JCL and procedures of DBRC, IMS batch applications,
   IMS utilities, and IMS tools.

   You must add DD statements to JCL and cataloged procedures for all the jobs
   that you want to activate Integrity Checker in. These JCL and procedures
   include those for DBRC, IMS batch applications, IMS utilities, and IMS Tools
   jobs that update IMS databases.

   **Important:** For integrity, ensure that all JCL and procedures that might change
   databases meet the following STEPLIB DD requirements.

   Add the following DD statements:

   **STEPLIB DD**
   Add the load module data sets that contain the global option module
   and the LIU load module library to the STEPLIB concatenation.
   The LIU load module library must be concatenated before the IMS load
   module library and must be APF-authorized.
   If you want Integrity Checker to detect changes in the logic of IMS user
   exits, which include randomizing routines, segment edit/compression
   exit routines, and HALDB or DEDB partition selection exit routines,
   also include the exit load modules in the STEPLIB concatenation.
   Integrity Checker does not check the user exit load modules in the LPA,
   ELPA, or LNKLST.

   **FABLPRNT DD**
   Optionally, you can specify the FABLPRNT DD statement in your
   procedures. This statement causes Integrity Checker to generate
   messages in the DD.
   If this statement is specified, Integrity Checker writes messages into
   this DD in addition to issuing the WTO macro. Each message contains
   a time stamp in its prefix, and you can easily identify the messages in
   relation to the authorization request from your application programs.

   **FABLSNAP DD**
   Optionally, you can specify the FABLSNAP DD statement in your
   online DBRC procedure. This statement causes Integrity Checker to
   generate diagnostic information for the VSAM control blocks when
   Integrity Checker gets a VSAM error. For DL/I batch jobs, you do not
   need to specify this DD statement because the jobs issue an abend
   dump when they get a VSAM error.

   **Example**

   The following figures show examples of the procedures.
Figure 10. Example of DBRC procedure for a non-BPE-based DBRC region

```
// PROC RGN=64M,DPTY='(14,15)',SOUT=A,
// IMSID=SYS3,SYS2=,IMSPLEX=
// IEFPROC EXEC PGM=DFSMVRC0,REGION=&RGN,
// DPTY=&DPTY,PARM='DRC,&IMSID,IMSPLEX=&IMSPLEX'
// ************************************************************************
//************
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR <<---- STEPLIB DD
//PROC LIB DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//JCLOUT DD SYSOUT=(A,INTDR)
//JCLPDS DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT
//SYSABEND DD SYSOUT=&SOUT
//FABLPRNT DD SYSOUT=&SOUT <<---- FABLPRNT DD
//FABLSNAP DD SYSOUT=&SOUT <<---- FABLSNAP DD
```

Figure 11. Example of DBRC procedure for a BPE-based DBRC region

```
// DBRC PROC RGN=OM,SOUT=A,
// RESLIB='IMS.SDFSRESL',
// BPECFG=BPECONFIG,
// DBRCINIT=000,
// IMSID=IMS1,
// PARM1=BPEINIT=DSBWIN0'
//*
// DBRCPROC EXEC PGM=BPEINIO0,REGION=&RGN,
// PARM='BPECFG=BPECFG,DRBCINIT=0,DRBCINIT,IMSID=IMS1,&PARM1'
//*
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR <<---- STEPLIB DD
//PROC LIB DD DSN=RESLIB,DISP=SHR
//PROC LIB DD DSN=SYS1.CSSSLIB,DISP=SHR
//SYSPRINT DD SYSOUT=&SOUT
//SYSUDUMP DD SYSOUT=&SOUT
//JCLOUT DD SYSOUT=(A,INTDR)
//JCLPDS DD DSN=SYS2.PROCLIB,DISP=SHR
//SYSABEND DD SYSOUT=&SOUT
//FABLPRNT DD SYSOUT=&SOUT <<---- FABLPRNT DD
//FABLSNAP DD SYSOUT=&SOUT <<---- FABLSNAP DD
```
Method 2. Customizing LIU load modules by merging into the IMS SDFSRESL library

Back up the IMS SDFSRESL library, and then merge the LIU load modules into the IMS SDFSRESL library.

Procedure

1. Back up the IMS SDFSRESL library.
   
   When program temporary fixes (PTFs) are released for the DSPCRTR0 module, you must restore the DSPCRTR0 module from the backup to apply the PTFs. Therefore, before merging LIU load modules, you must create a backup of the IMS SDFSRESL library. The backup is also required to deactivate Integrity Checker.

2. Use SMP/E to apply and accept IMS and IMS Library Integrity Utilities maintenance, and ensure that both are at the latest maintenance level.

3. Run the FABLUMD1 job that is in the SHPSJCL0 JCL library.

   This job updates the SMP/E CSI of IMS. It runs SMP/E RECEIVE/APPLY of USERMOD to install the FABLTR0 module into the IMS SDFSRESL library.
The FABLUMD1 job is shown in "JCL example to install the FABLRTR0 module into the IMS SDFSRESL library.”

4. Merge the LIU load modules (FABL* members) in the target library SHPSLMDO into the IMS SDFSRESL library.

If this step is not done, when Integrity Checker is activated, an ABENDU0109 load failure occurs for the required LIU load modules.

**JCL example to install the FABLRTR0 module into the IMS SDFSRESL library**

```jcl
//FABLUMD1 JOB
//---------------------------------------------------------------
//STEP1: SMP/E RECEIVE/APPLY usermod ZZLIU01 to IMS CSI
//---------------------------------------------------------------
//STEP1 EXEC procedure name of IMS SMP/E job
//
//+USERMOD(ZZLIU01) /* LIU R2 USERMOD */
//+REWORK(2004058) /* */
//+/* OPTIONAL LIU USERMOD FOR IMS-DBRC FMIDS. /* */
//+/* APPLY THIS USERMOD TO IMS-DBRC SMP/E CSI ONLY IF YOU WANT TO /* */
//+/* INSTALL LIU-MODIFIED VERSION OF DSPCRTR0 INTO YOUR IMS SMP/E /* */
//+/* CSI. /* */
//+/* +/ BEFORE APPLYING THIS USERMOD, ALL MAINTENANCE FOR IMS-DBRC /* */
//+/* DSPCRTR0 MUST BE ACCEPTED OR RESTORED. /* */
//+/* +/.
//+/*++VER(P115) /* IMS SYSTEM ID/FMID */
//+/* FMID(FMID of IMS) /* */
//+/* +/.
//+/*++JCLIN CALLLLIBS /* JCLIN FOR LIU MODULES */
//+/* +/.
//LINK EXEC PGM=IEWL,REGION=0M,
// PARM='SIZE=(880K,64K),LET,LIST,NCAL,RENT,REFR,XREF'
//SYSPRINT DD SYSOUT=A
//AHPSMODO DD DISP=OLD,DSN=LIU.AHPSMODO
//ADFSLOAD DD DISP=OLD,DSN=IMS.ADFSLOAD
//SYSMOD DD DISP=OLD,DSN=IMS.SDFSRESL
//SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(10,1))
//SYSLIN DD *
//INCLUDE AHPSMODO(FABLRT0)
//INCLUDE ADFSLOAD(DSPCRTR0)
//ENTRY FABLRTR0
//NAME DSPCRTR0(R)
//+
//MOD (FABLRTR0) LKLIB(AHPSMODO) /* LIU MODULE MOD ENTRY */
//@@
//SMPCTL DD *
//SET BDY (GLOBAL).
//RECEIVE S (ZZLIU01) SYMSMODS.
//SET BDY (TZONE name of IMS).
//APPLY S (ZZLIU01).
//*/
```
Configuring for a BPE-based DBRC environment

To activate Integrity Checker in an IMS online environment that has a BPE-based DBRC region, you must configure the members of the IMS PROCLIB data set. This step is required only when a BPE-based DBRC is used.

Procedure

1. Set up the BPE configuration parameter member.
   Specify the DBRC user exit list member by using the EXITMBR statement in the BPE configuration parameter member. The BPE configuration parameter member is specified by the BPECFG= keyword in the DBRC procedure that is used for the BPE-based DBRC region in which you want to activate Integrity Checker.

   You can skip this step if the EXITMBR statement for the DBRC user exit list member already exists.
   The following example specifies the EXITMBR statement in a BPE configuration parameter member:

   ```
   #User exit list PROCLIB member specification
   #EXITMBR=(member_name, DBRC) /* DBRC user exit list member */
   ``

   For more information about the EXITMBR statement, see the topic "BPE configuration parameter member of the IMS PROCLIB data set" in IMS System Definition.

2. Set up the DBRC user exit list member.
   Specify the Integrity Checker load module FABLBIN0 on the EXITDEF statement. The FABLBIN0 module must be specified as a DBRC Request exit in the DBRC user exit list member. The DBRC user exit list member is specified by the EXITMBR statement for DBRC in the BPE configuration parameter member.

   - When you have two or more DBRC user exits, the FABLBIN0 module must be specified as the first member on the EXITDEF statement.
   - Do not specify the ABLIM parameter on the EXITDEF statement.
   The following example specifies the Integrity Checker load module on the EXITDEF statement:

   ```
   *------------------------------------------------------------------
   # DBRC USER EXIT LIST PROCLIB MEMBER
   *------------------------------------------------------------------
   #DEFINE a DBRC request user exit.                           
   #EXITDEF(TYPE=REQUEST, EXITS=(FABLBIN0), COMP=DBRC)
   ```

   For information about the EXITDEF statement, see the topic "BPE exit list members of the IMS PROCLIB data set" in IMS System Definition.

Verifying that Integrity Checker is activated

IMS Library Integrity Utilities provides sample JCL for verifying successful activation of Integrity Checker. You can modify the sample JCL and then use it to ensure that Integrity Checker is running correctly.
About this task

This task is optional. Complete this task only if you want to ensure that Integrity Checker is activated.

Procedure

In the SHPSJCL0 library, locate sample JCL member FABLIVP3. Modify the sample JCL by following the instructions in the sample JCL and submit the JCL. Ensure that the job ends without errors.

Restarting IMS online and running IMS batch application, IMS utility, and IMS Tools jobs

When you have done all the steps, you are ready to restart IMS online to activate the DMB verification process.

Procedure

1. Restart IMS online and run IMS batch application jobs, IMS utility jobs, and IMS Tools jobs.
2. Confirm that the DMB verification process is activated by locating the following WTO message:
   
   FABL0114I LIU INTEGRITY CHECKER ACTIVATED. IMS VERSION IS version

Related reference:

“Output from Integrity Checker” on page 89
Maintaining Integrity Checker

To have Integrity Checker prevent database corruptions caused by using incorrect DBDLIBs, ACBLIBs, or RECON data sets, you must maintain the Integrity Checker resources appropriately.

The following topics explain the Integrity Checker maintenance tasks:

- “Maintaining RDEs”
- “Maintaining global option modules” on page 75
- “Maintaining LICON data sets” on page 77
- “Restarting Integrity Checker after an abend” on page 77
- “Applying PTFs to IMS Library Integrity Utilities and to IMS” on page 78

Important: Before you perform maintenance tasks on your databases, you must understand how Integrity Checker maintains RDEs and, if necessary, perform the manual operations that are required for such database maintenance tasks. Maintenance tasks in this context include the following tasks:

- Loading databases
- Reorganizing databases
- Recovering databases
- Changing DBDs
- Changing RECON records, for example, changing the maximum OSAM data set size for a HALDB

Maintaining RDEs

Integrity Checker automatically maintains the RDEs. However, with certain database maintenance tasks, you must use the LICON utility to manually re-create, delete, or expire RDEs.

Use the following topics to learn the maintenance tasks for RDEs that are required for each database maintenance task:

- “RDE maintenance at initial database load”
- “RDE maintenance at database reorganization” on page 68
- “RDE maintenance at database recovery” on page 70
- “RDE maintenance at DBD or RECON change” on page 71

RDE maintenance at initial database load

Integrity Checker stores the DMB information that IMS used to load the database in an RDE, and refers that DMB information as the correct DMB information.

When one of the following tools is used for the initial load of a database, Integrity Checker automatically creates an RDE for the database:

- IMS HISAM Reorganization Reload utility
- IMS HD Reorganization Reload utility
- IMS Database Reorganization Expert
  - IPR Reload Utility
  - Smart Reorg Driver with the REORGINPUT=ULDS option
- IMS High Performance Load
- IMS Online Reorganization Facility
- IMS Fast Path Advanced Tool of IMS Fast Path Solution Pack
If a database is initially loaded with another load application that runs with PROCOPT=L, Integrity Checker does not create an RDE; you must manually create an RDE by using the LICON utility. However, if an RDE does not exist for the database and the runtime option to automatically create an RDE is specified in the global option module (the RDEBUILD=Y option), Integrity Checker creates an RDE automatically.

**RDE maintenance at database reorganization**

Two types of reorganization are supported for IMS databases: reorganization without a DBD change and reorganization with a DBD change.

For a database reorganization without a DBD change, DMB verifications run while the database is being unloaded and reloaded by using the same DMB information. Because Integrity Checker uses the same RDE, the RDE does not need to be updated.

For a database reorganization with a DBD change, the first DMB verification is done while the database is being unloaded. For this DMB verification, Integrity Checker uses the DMB information before the DBD change. The second DMB verification is done while the database is being reloaded. For this DMB verification, Integrity Checker uses the DMB information that reflects the DBD change. Therefore, before the database is reloaded, the RDE must be re-created by using the updated DBD information.

If you use one of the following tools to reorganize the database, Integrity Checker automatically creates an RDE that contains the DMB information that reflects the DBD change:

- IMS HISAM Reorganization Reload utility
- IMS HD Reorganization Reload utility
- IMS Database Reorganization Expert
  - Smart Reorg utility
  - IPR Reload utility
- IMS High Performance Load
- IMS Online Reorganization Facility (without the ONLINECHANGE(N) option)
- IMS Fast Path Advanced Tool of IMS Fast Path Solution Pack

If you use other tools to reorganize the database, before you reload the database, use the INIT.DB command of the LICON utility to manually create an RDE by specifying the updated DBDLIB and the load library that contains the user exit routine as input to the utility.

If you use IMS Online Reorganization Facility with the ONLINECHANGE(N) option to reorganize the database, before you restart the database, use the INIT.DB command of the LICON utility to manually create an RDE by specifying the updated DBDLIB and the load library that contains the user exit routine as input to the utility.

**Considerations when reorganizing databases with IMS Database Reorganization Expert**

When you reorganize databases to change the DBD definition by using the Smart Reorg utility of IMS Database Reorganization Expert, Integrity Checker creates new RDEs for the changed databases. The following considerations pertain to the maintenance of RDEs.
When Integrity Checker fails to create RDEs

When Integrity Checker fails to create an RDE for a reason such as insufficient space in the LICON data set, it issues error messages. When you receive error messages, to secure the consistency of DMBs, you must confirm whether the RDEs contain the latest DMB information, and, if necessary, create new RDEs.

1. Check the time stamp of the RDEs.
   Run the LIST.DB command of the LICON utility by specifying the reorganized databases as input. From the listing that is generated by the job, check the time stamp to see when the RDE was created. If the time stamp matches the time of the reorganization, the latest RDE was created during the last reorganization.

2. Ensure that the reorganization of the databases completed successfully.
   Examine the messages and reports that are generated by the Smart Reorg utility of IMS Database Reorganization Expert to confirm that the reorganization of the databases completed successfully.

3. If the databases were reorganized successfully but RDEs are not the latest, create new RDEs.
   a. Resolve the cause of the error that occurred while creating new RDEs. For example, by defining a larger LICON data set and copying the data to the new LICON data set.
   b. Create new RDEs by using the INIT.DB command of the LICON utility and specifying the updated DBD library and the load library that contains the user exit routine as input. Alternatively, you can create new RDEs by running the reorganization job again.

When you restore the changed databases to their original state

If you decide not to use the reorganized database created with NAMESWAP=NO option or restore the databases and DBD definitions to the original state, you must also restore the RDEs. Run the RECOVER.DB command of the LICON utility.

To restore the RDE, both of the following requirements must be satisfied:
- Global option module specifies that historical copies of RDEs are kept (GENMAX=1 or higher).
- The RDE that was used before the DBD change is stored in the LICON data set as an expired RDE.

If the expired RDE does not exist, you must use the INIT.DB command of the LICON utility to manually create an RDE that contains the DMB information of the original state.

Considerations when reorganizing databases with IMS Fast Path

Advanced Tool of IMS Fast Path Solution Pack

When you reorganize DEDB areas to change the DBD definition by using the Change function or the combination of the Unload and Reload functions, Integrity Checker creates new RDEs for the changed areas. The following considerations pertain to the maintenance of RDEs.

When Integrity Checker fails to create RDEs

When Integrity Checker fails to create an RDE for a reason such as insufficient space in the LICON data set, it issues error messages. When you receive error messages, to secure the consistency of DMBs, you must confirm whether the RDEs contain the latest DMB information, and, if necessary, create new RDEs.
1. Check the time stamp of the RDEs.
   Run the LIST.DB command of the LICON utility by specifying the reorganized areas as input. From the listing that is generated by the job, check the time stamp to see when the RDE was created. If the time stamp matches the time of the reorganization, the latest RDE was created during the last reorganization.

2. Ensure that the reorganization of the areas completed successfully.
   Examine the messages and reports that are generated by IMS Fast Path Advanced Tool to confirm that the reorganization of the areas completed successfully.

3. If the areas were reorganized successfully but RDEs are not the latest, create new RDEs.
   a. Resolve the cause of the error that occurred while creating new RDEs. For example, by defining a larger LICON data set and copying the data to the new LICON data set.
   b. Create new RDEs by using the INIT.DB command of the LICON utility and specifying the updated ACB library and the load library that contains the user exit routine as input. Alternatively, you can create new RDEs by running the reorganization job again.

When you restore the changed areas to their original state
   If you restore the areas and DBD definitions to the original state, you must also restore the RDEs. Run the RECOVER.DB command of the LICON utility.

To restore the RDE, both of the following requirements must be satisfied:
   • Global option module specifies that historical copies of RDEs are kept (GENMAX=1 or higher).
   • The RDE that was used before the DBD change is stored in the LICON data set as an expired RDE.

If the expired RDE does not exist, you must use the INIT.DB command of the LICON utility to manually create an RDE that contains the DMB information of the original state.

Considerations when reorganizing databases with IMS Fast Path Basic Tools of IMS Fast Path Solution Pack

When you reorganize a DEDB area to change the DBD definition, Integrity Checker does not create an RDE for the reorganized area. Before you use the reorganized area, you must create an RDE for the area by using the LICON utility.

RDE maintenance at database recovery

Two recovery types are supported for IMS databases: recovery that recovers the database to the state that is defined by the current DBD, and recovery that recovers the database to a state before a DBD change (time stamp recovery).

When you recover the database to the state that is defined by the current DBD, Integrity Checker verifies the DMB by using the latest DMB information.

When you recover the database to the state before a DBD change with a time stamp recovery, Integrity Checker verifies the DMB by using the DMB information that was used when the database backup was created. Therefore, you must restore the RDE that contains the DMB information that was used when the backup was created.
If you use one of the following tools to recover the database, Integrity Checker automatically restores the RDE that was used when the backup was created:

- IMS High Performance Image Copy
- IMS Database Recovery Facility of IMS Recovery Solution Pack

However, to have Integrity Checker automatically restore the RDE, both of the following requirements must be satisfied:

- Global option module specifies that historical copies of RDEs are kept (GENMAX=1 or higher).
- The RDE that was used when the backup was created is stored in the LICON data set as an expired RDE.

If the expired RDE does not exist, use the INIT.DB command to manually create an RDE that contains the DMB information that was used before the DBD change.

When you run a time stamp recovery at a Remote Site Recovery (RSR) active site, you must recover the correct RDE at the RSR tracking site before running a recovery job at the RSR tracking site.

**Considerations when recovering databases with IMS Database Recovery Facility**

If you recover a database with the time stamp recovery function of IMS Database Recovery Facility and use old DBDs to rebuild index databases by using IMS Index Builder during the recovery, you must create RDEs for the old DBDs before rebuilding the index databases.

**RDE maintenance at DBD or RECON change**

If you change a DBD or a RECON record, you must have Integrity Checker make change in the RDE so that Integrity Checker uses the updated DMB information to verify the DMBs.

Except for cases where RDEs are created automatically during initial database load or database reorganization, whenever you change a DBD, re-create the RDE by using the INIT.DB command of the LICON utility to reflect the updated DBD information.

There might be cases where you want to roll back a DBD change and use the DBD that was used before the change. If you restore the DBD that was used before the change, Integrity Checker must refer to the DMB information that reflects the DBD before the change as the correct DMB information. For such cases, except for cases where RDEs are restored automatically during database recovery, restore the expired RDE manually by using the RECOVER.DB command of the LICON utility.

To restore the RDE, both of the following requirements must be satisfied:

- Global option module specifies that historical copies of RDEs are kept (GENMAX=1 or higher).
- The RDE that was used before the DBD change is stored in the LICON data set as an expired RDE.

If the expired RDE does not exist, you must use the INIT.DB command of the LICON utility to manually create an RDE that contains the DMB information that was used before the DBD change.
Altering the definition of a DMB verification-enabled online HALDB by using the HALDB alter function:

When you alter the definition of an online HALDB database that has DMB verification turned on by using the HALDB alter function of IMS, you must stop the DMB verification process and then restart the DMB verification process after the HALDB is altered.

About this task

When you alter the definition of an online HALDB database by using the INITIATE OLREORG command with the ALTER option and an online change command (which are both IMS commands), you must temporarily stop the DMB verification process. After the new definition is applied to the online database, restart the DMB verification process with the new RDEs that contain the new definition. Without these steps, the DMB verification process uses the old definitions to verify the DMBs. Therefore, the DMB verification process might deny database authorization requests that use correct IMS control blocks.

Note: The ALTER option is supported in IMS Version 13 and later.

The following figure shows the steps to alter the definition of an online HALDB database that has DMB verification turned on.

---

Figure 13. Steps to alter an online HALDB that has DMB verification turned on

---

IMS Library Integrity Utilities User's Guide
Procedure
1. Run the DBDGEN procedure with the new definition to update the DBD member in the DBD library with the new database structure.

2. Run the ACBGEN utility to update the ACB member that is in the staging ACBLIB.

3. Use the LICON utility of the Integrity Checker utility to stop the DMB verification process. To do so, issue the following LICON utility command:
   \begin{verbatim}
   CHANGE.DB DBD(dbname) CHECKON(N)
   \end{verbatim}

4. Apply the new definition to the online database by issuing the INITIATE OLREORG command with the OPTION(ALTER) parameter.
   IMS reads the staging ACB library and applies the changes to the online database.
   The ALTER option is supported only by the type-2 format of the command. For more information about the command, see the topic "INITIATE OLREORG command" in \textit{IMS Commands}.

5. Use the ACB member online change (OLC) function to activate the new ACB member.
   The new DBD definition is applied to the ACBLIB and is used to access the altered database.
   For more information about the ACB member online change function, see the topic "Changing or adding IMS.ACBLIB members online" in \textit{IMS System Administration}.

6. Use the LICON utility of the Integrity Checker utility to re-create the RDEs to reflect the new definition and start the DMB verification process. To do so, issue the following LICON utility command:
   \begin{verbatim}
   INIT.DB DBD(dbname) CHECKON(Y,D) REPLACE
   \end{verbatim}
   The DMB verification process starts and uses the new DBD definition.

Altering the definition of a DMB verification-enabled online DEDB by using the DEDB Alter utility (DBFUDA00):

When you alter the definition of an online DEDB database that has DMB verification turned on by using the DEDB Alter utility (DBFUDA00) of IMS, you must stop the DMB verification process and then restart the DMB verification process after the DEDB is altered.

About this task

When you use the DEDB Alter utility (DBFUDA00) to alter the definition of an online DEDB, you must temporarily stop the DMB verification process. After the new definition is applied to the online database, restart the DMB verification process with the new RDEs that contain the new definition. Without these steps, the DMB verification process uses the old definitions to verify the DMBs. Therefore, the DMB verification process might deny database authorization requests that use correct IMS control blocks.

Note: The DEDB Alter utility is supported in IMS Version 13 and later.

The following figure shows the steps to alter the definition of an online DEDB that has DMB verification turned on.
**Procedure**

1. Run the DBDGEN procedure with the new definition to update the DBD member in the DBD library with the new database structure.

2. Run the ACBGEN utility to update the ACB member that is in the staging ACBLIB.

3. Use the LICON utility of the Integrity Checker utility to stop the DMB verification process. To do so, issue one of the following LICON utility commands:

<table>
<thead>
<tr>
<th>Changed definition</th>
<th>LICON utility command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Altering a DEDB area</td>
<td>CHANGE.DB DBD(dbname) AREA(areaname) CHECKON(N)</td>
</tr>
<tr>
<td>Replacing the randomizer</td>
<td>CHANGE.DB DBD(dbname) CHECKON(N)</td>
</tr>
</tbody>
</table>

4. Run the DEDB Alter utility to apply the changes to the online DEDB. The new DBD definition is applied to the ACBLIB and is used to access the altered database.

   For more information about the DEDB Alter utility, see the topic "DEDB Alter utility (DBFUDA00)" in IMS Database Utilities.

5. Use the LICON utility of the Integrity Checker utility to re-create the RDEs to reflect the new definition and start the DMB verification process. To do so, issue one of the following LICON utility commands:
### Changed definition

<table>
<thead>
<tr>
<th>Changed definition</th>
<th>LICON utility command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Altering a DEDB area</td>
<td>INIT.DB DBD(dbname) AREA(areaname) CHECKON(Y,D) REPLACE</td>
</tr>
<tr>
<td>Replacing the randomizer</td>
<td>INIT.DB DBD(dbname) CHECKON(Y,D) REPLACE</td>
</tr>
</tbody>
</table>

The DMB verification process starts and uses the new DBD definition.

### Changing the maximum OSAM data set size for a DMB verification-enabled HALDB:

When you change the maximum size of OSAM data sets for a HALDB from 4 GB to 8 GB or 8 GB to 4 GB, you must use the LICON utility and manually create RDEs to reflect the change for the DMB verification process.

#### About this task

IMS supports, in Version 13 and later, the capability to change the maximum size of OSAM data sets for a HALDB from 4 GB to 8 GB or from 8 GB to 4 GB. When you change the maximum size of OSAM data sets for a HALDB, you must also re-create RDEs. If you do not re-create RDEs, the DMB verification process uses the old definition to verify the DMBs and the process might deny database authorization requests that use correct IMS control blocks.

#### Procedure

Follow the instructions in the topic “The maximum size of OSAM data sets and HALDB databases” in *IMS Database Administration* to change the maximum size of the data sets. After you issue the CHANGE.DB command with the OSAM8G keyword or the NOOSAM8G keyword, use the LICON utility of the Integrity Checker utility to re-create the RDEs for all of the partitions to reflect the new definition. To do so, issue the following LICON utility command:

```
INIT.DB DBD(haldb_master) REPLACE
```

Then, continue with the steps in *IMS Database Administration* to make the changes effective.

### Maintaining global option modules

You must maintain global options modules so that Integrity Checker applies appropriate values when creating RDEs. When multiple global option modules exist with different effective ranges, Integrity Checker uses its precedence rule to determine which options to apply in creating new RDEs.

See the following topics to maintain global option modules:

- “Changing the global option module”
- “Options applied to RDEs when multiple global option modules exist with different effective ranges” on page 76

### Changing the global option module

To change the behavior of Integrity Checker or the default options used for creating new RDEs, change the global option module.

#### About this task

Global option modules contain the options for controlling the behavior of Integrity Checker and options that are applied when creating new RDEs. There might be
times when you want to change these options.

**Procedure**

To change the global option module, delete it and create a new one. The global option values specified in the module are applied in creating new RDEs; replacing the module does not affect any existing RDEs. If you want the values in the new global option module to be applied to the existing RDEs, by using the DELETE.DB or EXPIRE.DB command of the LICON utility, delete or expire the existing RDEs that correspond to the global option module, and create new ones.

**Related tasks:**

“Setting up the global option modules” on page 54

**Related reference:**

“DELETE.DB command” on page 110

**Options applied to RDEs when multiple global option modules exist with different effective ranges**

The effective range of a global option module is either the installation level or IMS subsystem level. When multiple global option modules with different effective ranges exist, the option values applied when creating an RDE are inherited from multiple global option modules.

You create each global option module with the required effective range for your environment, but you can also create multiple global option modules in different effective ranges.

For example, you have multiple non-data-sharing IMS environments and you create a LICON data set for each environment, but you want to apply some common options. In this case, you can create one global option module for each IMS environment to apply unique options to each environment (global option module at the IMS subsystem level), and one global option module that contains common options to apply to all the IMS environments (global option module at the installation level).

When multiple global option modules exist with different effective ranges, Integrity Checker uses the following precedence rules to determine the options to apply in creating new RDEs. The same rules are also used when you create RDEs with the LICON utility.

1. **Values hardcoded in Integrity Checker**

   In all the five IMS environments (online IMS subsystems, batch jobs, user load programs, batch image copy jobs, and database recovery jobs), the default values for the verification options are hardcoded as follows:
   - DMB verification is done.
   - If a mismatch is found, Integrity Checker denies authorization to access the database.
   - No expired RDE are kept in the LICON data set (GENMAX=0).

2. **Global option module at the installation level**

   If Integrity Checker finds LIU@INST or LIUGINST in the execution libraries, it loads the global option module at the installation level and uses it. The values that you specified in LIU@INST or LIUGINST override the default values hardcoded in Integrity Checker.
If Integrity Checker finds both LIU@INST and LIUGINST, Integrity Checker ignores LIUGINST and uses LIU@INST.

3. Global option module at the IMS subsystem level
If Integrity Checker finds LIU@imsid or LIUGimsid that is associated with the IMS subsystem it runs in, it loads the global option module at the IMS subsystem level and uses it. The values that you specified in LIU@imsid or LIUGimsid override both the values in the global option module at installation level and the default values hardcoded in Integrity Checker.

If Integrity Checker finds both LIU@imsid and LIUGimsid, Integrity Checker ignores LIUGimsid and uses LIU@imsid.

4. FABLIN parameters that are provided in the LICON utility job run
If Integrity Checker finds any parameters that specify verification option values in the INIT.DB control statement of the LICON utility job (the job to create an RDE), those parameters override any other option values.

If you request Integrity Checker to automatically create an RDE during the first access to the database after the installation of Integrity Checker, Integrity Checker uses the option values in the preceding list except for the fourth one.

These precedence rules apply only for creating new RDEs, that is, when there are no existing RDEs. If an RDE exists, the verification option values that are set in the most recent RDE are carried over to the new RDE.

Maintaining LICON data sets
Because LICON data sets are KSDS data sets, CI/CA could split when Integrity Checker inserts a new RDE record into the LICON data set. Regular reorganization of the LICON data set helps you avoid frequent CI/CA splits.

Procedure

Tip: Frequent CI/CA splits might degrade performance. To avoid this, consider doing a batch registration of RDE, rather than having Integrity Checker create an RDE at the first opening of a database.

To reorganize a LICON data set, complete the following steps:
1. Define the output LICON data set with enough free space.
2. Copy the old LICON data set to the one defined by using the REPRO command of the VSAM access method services (IDCAMS).

   For more information about the REPRO command, see z/OS DFSMS Access Method Services for Catalogs.

Restarting Integrity Checker after an abend
When Integrity Checker terminates abnormally, the IMS online subsystems, batch applications, and utilities also terminate.

Procedure

Identify the cause of the error, correct the problem, and restart the IMS online applications, batch applications, or utilities.
If BPE-based DBRC is used, the IMS online subsystem does not terminate when Integrity Checker abnormally terminates. To reactivate Integrity Checker, after you correct the error, restart the IMS subsystem. Do not issue a BPE USEREXIT command until the IMS subsystem is restarted.
Applying PTFs to IMS Library Integrity Utilities and to IMS

If the Integrity Checker modules are merged into the IMS SDFSRESL library, in addition to the standard SMP/E steps, extra steps are required for applying PTFs.

Procedure

Complete the following steps when you apply PTFs:

- For IMS Library Integrity Utilities, after you apply a PTF, merge the members again.
- For IMS, if the PTF updates the DSPCRTR0 module, restore the DSPCRTR0 module before you apply the PTF. After you apply the PTF, create a backup of the IMS SDFSRESL data set, and then link-edit FABLSTR0 and DSPCRTR0 again.

After you apply the PTFs, complete either of the following steps depending on the DBRC environment used:

- For non-BPE based DBRC, restart the IMS online subsystems.
- For BPE-based DBRC, issue the BPE REFRESH USEREXIT command to reload the load modules of Integrity Checker. IMS online subsystems do not need to be restarted.

For more information about the BPE REFRESH USEREXIT command, see the topic "BPE REFRESH USEREXIT command" in IMS Commands.
Preventing database corruption with Integrity Checker

When Integrity Checker is in operation, it alerts you whenever a DMB mismatch is found. When a problem is reported by Integrity Checker, you must identify the cause, determine the action, and correct the problem to prevent database corruptions.

The following topics explain how to address a DMB mismatch:

• “Restrictions: Cases where DMB verification is not done”
• “DMB mismatch in IMS online environment or application jobs” on page 80
• “DMB mismatch during database maintenance and operation” on page 83
• “Addressing a DMB mismatch” on page 84

Restrictions: Cases where DMB verification is not done

DMB verification is not done for IMS batch programs, IMS utility programs, or IMS Tools programs when these jobs are run with DBRC inactive (DBRC=N). Also, even when DBRC is active (DBRC=Y), DMB verification is not done when certain tools are used or under certain conditions.

• Even when DBRC is active, DMB verification is not done for the following utility jobs:
  
  **IMS standard utilities**
  
  - Database Prefix Resolution utility (DFSURG0)
  - MSDB to DEDB Conversion utility (DBFUCDB0)
  - Database Prereorganization utility (DFSURPR0) (for non-HALDBs)
  - Database Change Accumulation utility (DFSUCUM0)
  - Batch Backout utility (DFSDB000)
  - HISAM ReorganizationUnload utility (DFSURUL0)
  - DEDB Initialization utility (DFSUMIN0)
  - Database Image Copy utility (DFSUDMP0) (for HALDBs)
  - Database Image Copy2 utility (DFSUDMP0) (for HALDBs)

  **IMS Tools products**
  
  - IMS High Performance Prefix Resolution
  - IMS High Performance Pointer Checker (for HALDBs)
  - IMS Fast Path Advanced Tool and IMS Fast Path Basic Tools of IMS Fast Path Solution Pack

  • To activate DMB verification in IMS Index Builder jobs, APAR PM53350 must be applied to IMS Index Builder, and IMS Index Builder JCL must specify DBAUTH YES (default). Otherwise, DMB verification is not done.

  • DMB verification cannot be used in IMS Fast Path Advanced Tool and IMS Fast Path Basic Tools jobs of IMS Fast Path Solution Pack.
  - DMB verification is supported in IMS Fast Path Online Tools jobs.
  - Integrity Checker can update RDEs and record time stamps when it is called in IMS Fast Path Advanced Tool jobs.

  • When a database authorization request is made by the DBRC application programming interface (API), Integrity Checker does not verify the DMB.

  • In an XRF environment, the alternate subsystem inherits databases and their authorization from the active subsystem during takeover. Integrity Checker does not verify the DMBs in the alternate subsystem (that is, the new active
subsystem) until the databases are stopped by the /STOP or the /DBR command, and then restarted by the /START command.

- In Fast Database Recovery (FDBR) regions, FDBR does not require authorization for any databases to be recovered. Consequently, Integrity Checker does not verify the DMBs in FDBR regions.

**DMB mismatch in IMS online environment or application jobs**

Integrity Checker verifies various information in DMBs. Therefore, before you address a DMB mismatch, you must know which elements are verified in DMB verification, how Integrity Checker verifies the DMBs, and the behavior of Integrity Checker when it detects a mismatch.

**Subsections:**
- “Elements verified in DMB verification”
- “DMB verification methods” on page 81
- “Actions when a DMB mismatch occurs” on page 81

**Elements verified in DMB verification**

Integrity Checker stores the DMB information that IMS used to load the database in the RDE, and refers to that information as the correct DMB information. Then, when the DMB verification process starts, it uses that information to verify whether the DMB in the DBDLIB or the ACBLIB referred to by the IMS application is correct.

Integrity Checker compares the following information against the information that is stored in the RDE. When a mismatch is found, Integrity Checker determines that an incorrect DMB is used.

**Information defined in the DBD**

- Compares database definition information that is defined in the DBD. However, the following elements are not compared:
  - Definitions that are not related to database structure
  - Field information

**Information defined in the RECON data sets**

- Compares the HALDB partition information and HALDB OSAM data set size information that is defined in the RECON data sets.

**Logic of the user exit routine**

DMB information includes the name of the user exit routine. Integrity Checker checks for changes in the logic of that user exit routine.

Integrity Checker calculates the checksum value of the user exit routine and stores it in the RDE. When DMB verification starts, Integrity Checker calculates the checksum of the user exit routine in the STEPLIB, and compares it with the value stored in the RDE. Even if the user exit routine is in an LPA or LINKLST, if you want to check for changes in the logic of the user exit routine, you must specify the library that contains the user exit routine in the STEPLIB.

This check is supported for the following user exit routines. Whether to verify each of these user exit routines can be requested individually by specifying the verification option.

- Randomizing routine
- Segment edit/compression exit routine
• HALDB partition selection exit routine
• DEDB partition selection exit routine

Version ID in the DBD

Compares the version ID in the DBD. Version ID is the 13-character time stamp of when the DBD was created or the character string that is specified on the VERSION keyword of the DBD statement that was supplied for DBDGEN. If the VERSION keyword was not specified, the version ID contains the time stamp of when the DBD was generated.

Unlike the other elements, the version ID is verified only when double-step verification is requested as the DMB verification method. The value is compared in the first step of double-step verification, and when a mismatch is found, the DMB verification process proceeds to the next step to verify other elements.

Even when a mismatch is found between the version IDs, Integrity Checker does not determine a version ID mismatch as a DMB mismatch.

DMB verification methods

For IMS full-function databases, DMB verification supports two methods: single-step verification and double-step verification. The global option module specifies which method to apply. When a method is not specified, single-step verification is applied. For DEDBs, single-step verification is always applied.

Single-step verification

Integrity Checker verifies all elements at once. This method is more reliable than the double-step method, but slower.

Double-step verification

Integrity Checker verifies the version ID of the DMB in the first step. If a mismatch is found between the version IDs, Integrity Checker proceeds to the next step and verifies the other elements. This method is faster than the single-step method, but less reliable.


Actions when a DMB mismatch occurs

The behavior of Integrity Checker when it detects a DMB mismatch is controlled by the DMB verification option. By using the DMB verification option, you can request either of the following behaviors as the action taken:

• Deny authorization
• Issue a warning and continue

If neither is specified, Integrity Checker denies authorization.

Deny authorization

When the DMB verification option specifies to deny authorization or when the DMB verification option is not specified, Integrity Checker prevents database corruption when it detects a DMB mismatch.

Integrity Checker intercepts DBRC authorization processing and verifies the DMB that is being used. If it detects a mismatch, it returns a non-zero return code and reason code $$ to the DBRC authorization requester. When the requester receives these codes, the requester perceives that DBRC
authorization failed and terminates database processing. Because the access to the database with an incorrect DMB is avoided, the database is safe from corruption.

When the requester receives the DBRC authorization failure notification, the requester issues the following message:

```
DFS047A - UNABLE TO OBTAIN AUTH. RSN=$$
```

After issuing this message, BMP, MPP, or IFP region that runs in the IMS online environment or the batch utility job abends with ABENDU0047, ABENDU3303, or with another abend code.

Integrity Checker also issues the following error messages.

- When a DMB mismatch is found, Integrity Checker issues the following error messages to indicate which value in the DMB differs from the value in the RDE:
  
  ```
  FABL0204E DMB MISMATCH FOUND FOR DBD: dbdname
  FABL0204E Field name in the DMB
  FABL0204E RDE VALUE: the value in RDE
  FABL0204E ACB VALUE: the value in ACB
  ```

- When a mismatch is found in the user exit routine checksum value, Integrity Checker issues the following error messages to indicate which user exit routine checksum differs from the checksum in the RDE:

  ```
  FABL0209E CHECKSUM MISMATCH FOUND FOR DBD: dbdname
  FABL0209E Type of the user exit routine
  FABL0209E MODULE NAME: name of the user exit routine
  ```

**Issue a warning and continue**

When the DMB verification option specifies to issue a warning and continue, even when a DMB mismatch is found, Integrity Checker allows the requester to access the database.

**Attention:** When this option is specified, even when a mismatch is found between the RDE and the DMB, Integrity Checker does not restrict database access. Use this option only when you intend to change the DBD and you want Integrity Checker to use the updated DMB information as the correct DMB information.

For example, user load applications match this case. So when you want to do an initial load the database with a user load application, use this DMB verification option.

If you specify this option when you do not intend to change the DBD, and if an incorrect DBDLIB or ACBLIB is used, Integrity Checker allows database access, and the database access might lead to a database corruption.

- When a DMB mismatch is found, Integrity Checker issues the following warning messages to indicate which value in the DMB differs from the value in the RDE:

  ```
  FABL0203W DMB MISMATCH FOUND FOR DBD: dbdname
  FABL0203W Field name in the DMB
  FABL0203W RDE VALUE: the value in RDE
  FABL0203W ACB VALUE: the value in ACB
  ```

- When a mismatch is found in the user exit routine checksum value, Integrity Checker issues the following warning messages to indicate which user exit routine checksum differs from the checksum in the RDE:
DMB mismatch during database maintenance and operation

Integrity Checker might report a DMB mismatch during database maintenance tasks or while you operate on the databases.

Important: When you perform database maintenance tasks, RDEs must also be maintained. For RDE maintenance tasks that are required for each database maintenance task, see “Maintaining RDEs” on page 67.

Subsections:
- “Initial database load”
- “Database reorganization”
- “Database recovery” on page 84
- “DBD change” on page 84

Initial database load

If a DMB mismatch is found during the initial load of the database, use the LICON utility to create an RDE by specifying the DBDLIB and the load library that contains the user exit routine that is used for the initial load, and redo the initial load of the database.

In certain circumstances, Integrity Checker automatically creates an RDE and does not verify the DMBs. For more information about how Integrity Checker maintains RDEs during initial database load, see “RDE maintenance at initial database load” on page 67.

Database reorganization

For a database reorganization that does not accompany a DBD change, DMB verifications run while the database is being unloaded and reloaded. If a DMB mismatch is found, it means that an incorrect DBDLIB or a load library that contains incorrect user exit routine is used. To resolve the problem, see “Addressing a DMB mismatch” on page 84.

For a database reorganization that accompanies a DBD change, the first DMB verification is done while the database is being unloaded. For this DMB verification, Integrity Checker uses the DMB information before the DBD change. The second DMB verification is done while the database is being reloaded. For this DMB verification, Integrity Checker uses the DMB information after the DBD change. If a DMB mismatch is found, it indicates that either a new RDE was not created or an incorrect DBDLIB or a load library that contains incorrect user exit routine is used.
routine is used. Ensure that a new RDE is created before database reload. For a
DMB mismatch caused by other errors, see “Addressing a DMB mismatch.”

In certain circumstances, Integrity Checker automatically creates an RDE. For more
information about how Integrity Checker maintains RDEs during database
reorganization, see “RDE maintenance at database reorganization” on page 68.

**Database recovery**

When you recover the database to the state that is defined by the current DBD,
Integrity Checker verifies the DMB by using the latest DMB information. If a DMB
mismatch is found, it means that an incorrect DBDLIB or a load library that
contains incorrect user exit routine is used.

If a DMB mismatch is found during a recovery to a state before a DBD change
(time stamp recovery), it means that the correct RDE is not restored or an incorrect
DBDLIB or a load library that contains incorrect user exit routine is used. Ensure
that the correct RDE is restored before recovering the database.

In both cases, for a DMB mismatch caused by other errors, see “Addressing a DMB
mismatch.”

In certain circumstances, Integrity Checker automatically restores the RDE. For
more information about how Integrity Checker maintains RDEs during database
recovery, see “RDE maintenance at database recovery” on page 70.

**DBD change**

If a DMB mismatch is found after a DBD change, ensure that an incorrect DBDLIB,
incorrect RECON data sets, or a load library that contains incorrect user exit
routine is not used. Also, ensure that the DBD change was done with appropriate
procedures, and that the DBDLIB, the RECON data sets, or the user exit routine
was regenerated to apply the change. If these steps are not done yet, perform the
steps.

If a DMB mismatch is found after restoring a DBD change, ensure that an incorrect
DBDLIB, incorrect RECON data sets, or a load library that contains incorrect user
exit routine is not used. Also, ensure that the DBD was restored with appropriate
procedures, and that the DBDLIB, the RECON data sets, or the user exit routine
was regenerated or restored to roll back the change. If these steps are not done yet,
perform the steps.

In both cases, for a DMB mismatch caused by other errors, see “Addressing a DMB
mismatch.”

**Addressing a DMB mismatch**

When Integrity Checker notifies you about a mismatch, it means that an incorrect
DBDLIB, ACBLIB, or RECON data sets are used or a load library that contains
incorrect user exit routine is used.

When a DMB mismatch is reported, investigate the cause. The cause can be, for
example, errors in JCL modification or in re-creation of data sets. After you identify
the cause, specify the correct data set and rerun the IMS job.
If a DBD change was made before the run, ensure that the DBD change was done with appropriate procedures, and that the DBDLIB, the ACBLIB, the RECON data sets, or the user exit routine is regenerated to apply the change. If these steps are not done, perform the steps.

You can use the following methods to identify the cause of the DMB mismatch and address the problem:

**Search for the correct DBDLIB, ACBLIB, or RECON data sets that match the RDE**

By using the VERIFY.DB command of the LICON utility, you can verify the DBDLIB, the ACBLIB, or the RECON data sets against the DMB information in the RDE.

If you have backups of the DBDLIB or the ACBLIB, you can compare them against the RDE and identify the DBDLIB or the ACBLIB that matches the RDE. If one of the backups match the RDE, you can use that DBDLIB backup or ACBLIB backup as the correct DBDLIB or ACBLIB.

If you maintain several sets of RECON data sets, you can check them against the RDE and identify whether those RECON data sets match the RDE. If a set of RECON data sets matches the RDE, you can use that set of RECON data sets as the correct RECON data sets.

**Identify and correct the mismatching element in DBDLIB, ACBLIB, or RECON data sets**

Identify the mismatching element in the DBDLIB, the ACBLIB, or the RECON data sets. The content of the RDE can be printed by using the LIST.DB command of the LICON utility, and the content of the DBDLIB, the ACBLIB, or the RECON data sets can be obtained by either of the following methods:

- Use the DBD/PSB/ACB Reversal function to print the content of the DBDLIB or the ACBLIB.
- From a web browser, use the DBD/PSB Map Viewer of Management Console to view the content of the DBDLIB.
- Use the DBRC LIST.DB command to print the content of the RECON data sets.

By comparing the two, correct the mismatching element in the DBDLIB, the ACBLIB, or the RECON data sets. Then, run DBDGEN, PSBGEN, ACBGEN, or issue a DBRC command and rerun the IMS job.

**Search for the load library that contains the correct user exit routine**

You can check whether the user exit routine is the correct routine by running the VERIFY.DB command of the LICON utility. For the STEPLIB of the LICON utility job, specify the load library that contains the user exit routine to check.

If you have multiple load libraries that each contains user exits, you can search for the correct load library by concatenating the libraries to the STEPLIB and running the VERIFY.DB command.

If the user exit routine used varies depending on the order of the load libraries concatenated to STEPLIB, then change the order of concatenation and run the VERIFY.DB command to identify the correct order of concatenation. After you identify the correct order of concatenation, update the JCL statements.
Deactivating Integrity Checker

Deactivate Integrity Checker if you want to stop using the DMB verification function.

About this task

The DMB verification process is activated when all of the following conditions are met:

• One or more global option modules exist.
• Alias name DSPCRRTR0 is defined for the LIU FABLRT0 load module or Integrity Checker modules are merged into the IMS SDFSRESL library.
• (When used in a BPE-based DBRC environment) DBRC user exit list member is modified for Integrity Checker.

To turn off Integrity Checker, complete one of the following procedures, depending on how IMS Library Integrity Utilities is installed in your environment.

If the LIU load module data set is merged into the IMS SDFSRESL library, the following procedures cannot be applied. You must restore the IMS SDFSRESL library from the backup or reinstall the IMS SDFSRESL library.

Procedure

Complete either of the following tasks to deactivate Integrity Checker:

• “Deactivating Integrity Checker when IMS Library Integrity Checker is installed as a stand-alone product” on page 87
• “Deactivating Integrity Checker when IMS Library Integrity Utilities is installed as a component of an IMS tools solution pack” on page 87

Deactivating Integrity Checker when IMS Library Integrity Checker is installed as a stand-alone product

If IMS Library Integrity Utilities is installed as a stand-alone product (that is, not through IMS Tools solution packs), the product target load module data set (SHPSLMD0) contains only the LIU modules. In such an environment, complete the following steps to deactivate Integrity Checker.

Before you begin

If multiple IMS Tools product target libraries are contained in a single data set, instead of completing the following steps, complete the steps in “Deactivating Integrity Checker when IMS Library Integrity Utilities is installed as a component of an IMS tools solution pack” on page 87 to deactivate Integrity Checker.

Procedure

1. Remove the product load module data set and the libraries that contain the global option modules from the STEPLIB concatenation in DBRC JCL, IMS batch application JCL, IMS utility JCL, and IMS Tools JCL.
2. If you use BPE-based DBRC, configure the member of the IMS PROCLIB data set to remove the FABLBIN0 module name from the EXITDEF statement of the DBRC user exit list.

For more information about the EXITDEF statement, see the topic "BPE exit list members of the IMS PROCLIB data set" in IMS System Definition.
3. Complete either of the following steps depending on the DBRC environment used:

- For non-BPE based DBRC, restart the IMS online subsystems.
- For BPE-based DBRC, issue the BPE REFRESH USEREXIT command to deactivate Integrity Checker. IMS online subsystems do not need to be restarted.

For more information about the BPE REFRESH USEREXIT command, see the topic "BPE REFRESH USEREXIT command" in *IMS Commands*.

**Deactivating Integrity Checker when IMS Library Integrity Utilities is installed as a component of an IMS tools solution pack**

If IMS Library Integrity Utilities is installed as a component of an IMS tools solution pack, all the product load modules are contained in the same data set. In such an environment, complete the following steps to deactivate Integrity Checker.

**Procedure**

1. Delete alias name DSPCRTR0, which is defined to the LIU FABLRTR0 load module. You can delete the alias name by running the IEHPROGM program. You can use the JCL example in "JCL example to remove alias name DSPCRTR0" to run this step. When the job ends, confirm that the return code is zero.

2. Remove the global option module from the STEPLIB concatenation in DBRC JCL, IMS batch application JCL, IMS utility JCL, and IMS Tools JCL.
   
   If the global option module is created in the load module library of IMS Database Solution Pack or IMS Fast Path Solution Pack, delete the global option module.
   
   If an alias name is defined for the global option module, delete the alias.

3. If you use BPE-based DBRC, configure the member of the IMS PROCLIB data set to remove the FABLBIN0 module name from the EXITDEF statement of the DBRC user exit list. Then, issue the BPE REFRESH USEREXIT command to deactivate Integrity Checker.
   
   See the following topics for additional information:
   
   - For the EXITDEF statement, see the topic "BPE exit list members of the IMS PROCLIB data set" in *IMS System Definition*.
   - For the BPE REFRESH USEREXIT command, see the topic "BPE REFRESH USEREXIT command" in *IMS Commands*.

**JCL example to remove alias name DSPCRTR0**

Use the following JCL example to remove alias name DSPCRTR0. This JCL is in the SHPSJCL0 library, member FABLALSD.
//FABLALSD JOB
/**-------------------------------------------------------*/
//**   STEP1: Scratch the alias DSPCRTC0 from SHPSLMD0
/**-------------------------------------------------------
//STEP1  EXEC PGM=IEHPROGM
/**
/**
/**SYSPRINT DD SYSOUT*
//liuvol DD DISP=SHR,UNIT=SYSALLDA, VOL=SER=liuvol
//SYSIN DD *
  SCRATCH MEMBER=DSPCRTC0, VOL=SYSALLDA=liuvol, C
    DSNNAME=LIU.SHPSLMD0 LIU target load module lib
/**

Figure 15. Removing the alias name DSPCRTC0
Output from Integrity Checker

Output from the Integrity Checker consists of the FABLPRNT data set and the FABLSNAP data set.

FABLPRNT data set

The FABLPRNT data set, which is an optional data set, contains messages issued by Integrity Checker.

If a FABLPRNT DD statement is specified in your procedure, Integrity Checker prints messages in this data set. The messages generated in this data set are the same as the WTO messages. Each message contains a time stamp in its prefix, and you can easily identify the messages in relation to the authorization request from your application programs.

The following figures show messages that are generated in the FABLPRNT data set.

The following messages are printed when no mismatches are found or when the verification option is set to (N), which means the verification option is turned off.

```
15086 20:09:58.63  FABL0101I LIU INTEGRITY CHECKER NOW ACTIVE WITH LICON: HLQ.IMS1.LICON
15086 20:09:58.63  FABL0102I LIU INTEGRITY CHECKER INITIALIZATION COMPLETED
15086 20:09:58.63  FABL0114I LIU INTEGRITY CHECKER ACTIVATED. IMS VERSION IS 13
```

Figure 16. Messages when no mismatches found or the verification option is (N)

The following messages are printed when the verification option is turned on and a mismatch is found. In this example, the verification option is set to (Y,W), which requests Integrity Checker to issue a warning message and create a new RDE when a mismatch is found.

```
15120 14:32:15.42  FABL0101I LIU INTEGRITY CHECKER NOW ACTIVE WITH LICON: HLQ.IMS1.LICON
15120 14:32:15.42  FABL0102I LIU INTEGRITY CHECKER INITIALIZATION COMPLETED
15120 14:32:15.42  FABL0114I LIU INTEGRITY CHECKER ACTIVATED. IMS VERSION IS 13
15120 14:32:18.61  FABL0203W DMB MISMATCH FOUND FOR DBD: DBTEST1
15120 14:32:18.61  FABL0203W NUMBER OF SEGMENT TYPES
15120 14:32:18.61  FABL0203W RDE VALUE:  3
15120 14:32:18.61  FABL0203W ACB VALUE:  2
15120 14:32:18.61  FABL0205E VERIFICATION PROCESS FOR DBTEST1 HAS BEEN STOPPED
15120 14:32:18.61  FABL0205E REASON: SEVERE DMB MISMATCH FOUND
15120 14:32:18.62  FABL0201I RDE CREATED FOR DBD: DBTEST1
```

Figure 17. Messages when the verification option is (Y,W) and a mismatch is found

The following messages are printed when the verification option is turned on and a mismatch is found. In this example, the verification option is set to (Y,D), which requests Integrity Checker to deny database authorization when a mismatch is found.
The FABLSNAP data set, which is an optional data set, contains diagnostic information about the VSAM control blocks. This data set is used only when Integrity Checker encounters a VSAM error.
Global option module generation macro

Use the global option module generation macro to create global option modules.

Creating global option modules

To activate Integrity Checker, create at least one global option module that contains the name of the LICON data set. If you want to change the default options of Integrity Checker globally, specify them when you create this module.

About this task

IBM does not supply global option modules. You must create at least one global option module before invoking Integrity Checker.

Procedure

To create a global option module, determine the type of the global option module, then use the FABLPGEN procedure (provided in the SHPSSAMP data set) to create it. Runtime options can be defined by using the SYSIN control statements. There are two levels of global option modules that can be categorized by their effective range: installation level and IMS subsystem level.

Installation level

To set values that will be effective at the installation level, create a global option module named LIU@INST. The values in this module apply to all the databases that are defined in the IMS environment.

IMS subsystem level

To set values that will be effective at the IMS subsystem level, create a global option module named LIU@imsid, where imsid is the 4-character ID of the IMS subsystem. The values that you set in this module apply to all the databases that are defined to that IMS subsystem. In accordance with the options assignment rule, they override the values that are set in the LIU@INST module.

Tip: If you want more than one IMS subsystems to use a set of options that are defined in a single global option module, create a global option module for the IMS subsystem level and use the linkage editor to assign an alias to that global option module.

In a database sharing environment where more than one IMS subsystem shares databases, the LICON data set and option values defined in the global option module must be the same across the IMS subsystems. Assigning an alias is beneficial in such a case as well as in XRF environments. For information about how to assign an alias name, see “Setting up the global option modules” on page 54.

In environments where a LICON data set is used across multiple IMS subsystems, assigning an alias name for the global option module to apply the same runtime options for all IMS IDs is a good practice. However, if you want to set runtime options for each IMS ID, instead of assigning an alias name to the global option module, you can create one global option module for each IMS ID. In such a case, except for certain control statement keywords, the keyword parameters must be the same. You can set different parameters for the following control statement keywords:

• VERIFY=
When you create global option modules, name the modules LIU@xxxx. At sign (@) is a code-page-dependent character. If you are working in an environment where you cannot use the at sign (@), name the modules LIUGxxxx. Use either format for all the global option modules consistently because maintaining both LIU@xxxx and LIUGxxxx modules can cause confusion. If Integrity Checker finds both LIU@imsid and LIUGimsid in the same effective range level, Integrity Checker ignores LIUGimsid and uses LIU@imsid.

Related concepts:
“Options applied to RDEs when multiple global option modules exist with different effective ranges” on page 76

JCL requirements for the FABLPGEN program

The following JCL requirements must be met to create a global option module with the FABLPGEN program.

Subsections:
- “EXEC statement”
- “DD statements”
- “Control statement keywords” on page 93
- “Example” on page 97

EXEC statement

The EXEC statement must be in the following form.

```
//stepname EXEC FABLPGEN,MBR=module,SOUT=x
```

**MBR=** Specifies the name of the global option module. *module* is LIU@INST, LIUGINST, LIU@imsid, or LIUGimsid.

**SOUT=**

Specifies the SYSOUT class to be used for SYSPRINT DD.

DD statements

Code the following DD statements to identify the source of input and the placement of output information:

**SYSLIB DD**

This statement specifies the macro library (SHPSMAC0) provided by IMS Library Integrity Utilities, or one of IMS tools solution packs. This library contains the FABLPGIN macro.

**SYSIN DD**

This statement specifies the input control statement stream.

**SYSLMOD DD**

This statement specifies the output data set for global option modules. If you merge the Integrity Checker load modules into the IMS SDFSRESL
library, specify the IMS SDFSRESL library for this DD statement. Otherwise, specify the Integrity Checker load module library for this DD statement.

You can create global option modules in a different library. If you do so, concatenate that library to the STEPLIB DD in JCL and cataloged procedures for all the jobs from which you want to activate Integrity Checker.

**Control statement keywords**

The control statement formats are as follows:

**FABLPGIN**

The IBM supplied macro for use in defining the global option module. The syntax of the parameter specifications of this macro is the same as the syntax of an ordinary assembler macro statement.

You must specify the statement label for the FABLPGIN macro. For the statement label, specify the name of the global option module, which is LIU@INST, LIUGINST, LIU@imsid, or LIUGimsid (in the example in Figure 19 on page 97 LIU@INST beginning at column 1.)

**LICON=**

Specifies the name of the LICON data set. No system default value is provided for this parameter.

**VERIFY=**

Specifies the method for verifying the DMBs; either SNGL or DBLE. SNGL specifies single-step verification and DBLE specifies double-step verification. The system default value for this parameter is SNGL.

You can choose either of the following two options for how Integrity Checker verifies the DMB of a full-function database against the DMB information registered in the RDE. Specify your choice in the global option module.

**Single-step verification**

Integrity Checker verifies all elements at once. This method is more reliable than the double-step method, but slower.

**Double-step verification**

Integrity Checker verifies the version ID of the DMB in the first step. Version ID is the 13-character time stamp of when the DBD was created or the character string that is specified on the VERSION= keyword of the DBD statement that was supplied for DBDGEN. If the version IDs are not the same, Integrity Checker proceeds to the next step to verify other elements. This method is faster than the single-step method, but less reliable.

For DEDBs, single-step verification is always applied.

**Recommendation:** Specify single-step verification. If you experience a performance problem, consider using double-step verification.

**MSGROUT=**

Specifies the message routing codes for write-to-operator (WTO) messages issued by Integrity Checker. You can specify values in the range of 1 - 16. The system default value for this parameter is (2,7,11).
MSGDESC=
Specifies the message descriptor codes for write-to-operator (WTO) messages issued by Integrity Checker. The system default value for this parameter is (7).

VERIFYLMT=
Specifies the maximum number of the mismatch messages to be issued for a DMB. For example, specifying 3 means Integrity Checker does not issue more than three mismatch messages for a DMB.

You can specify any number in the range of 0 - 99. 0 specifies that verification is to be done but no mismatch message issued. 99 specifies that the number of messages is unlimited. The system default value for this parameter is 10.

RDEBUILD=
Specifies whether Integrity Checker automatically creates an RDE. The system default value for this parameter is Y.

Y If no current RDE exists for a DEDB area, a non-HALDB full-function database, or a HALDB partition, Integrity Checker automatically creates an RDE during the first access to it.

N Even when no current RDE exists for a DEDB area, a non-HALDB full-function database, or a HALDB partition, Integrity Checker does not create an RDE during the first access to it.

INITERR=
Specifies whether Integrity Checker abnormally ends, or issues a warning message and stops its processing, when the initialization of Integrity Checker fails. The system default value for this parameter is A.

This option is not effective in an IMS online subsystem that has a BPE-based DBRC region. If you specify this option for such an environment, Integrity Checker stops processing, and the IMS online subsystem continues processing.

A If the initialization of Integrity Checker fails, it ends abnormally together with the IMS online subsystem that has a non-BPE-based DBRC region or the IMS batch job.

W If the initialization of Integrity Checker fails, it issues a warning message and stops its processing. The IMS online subsystem or the IMS batch job continues processing without the Integrity Checker function. However, Integrity Checker ends abnormally if errors occur before the effective value for this option is decided. Such errors are load failures of the following modules:
• FABLRTRx (x: 8, 9, A, B, C, or D)
• FABLWM0
• FABLA10
• Global option module

CHECKON=
Specifies the verification option for online IMS subsystems. The system default value for this parameter is (Y,D). You can specify the following options:

Y Check. If you specify Y, you can specify either of the following parameters:
If a mismatch is found, deny authorization to use the database.

Issue a warning message and create a new RDE.

Do not check.

CHECKBAT=
Specifies the verification option for batch jobs. The system default value for this parameter is (Y,D). You can specify the following options:

Y Check. If you specify Y, you can specify either of the following parameters:

D If a mismatch is found, deny authorization to use the database.

W Issue a warning message and create a new RDE.

N Do not check.

CHECKLD=
Specifies the verification option for user load program jobs. The system default value for this parameter is (Y,D). You can specify the following options:

Y Check. If you specify Y, you can specify either of the following parameters:

D If a mismatch is found, deny authorization to use the database.

W Issue a warning message and create a new RDE.

N Do not check.

CHECKIC=
Specifies the verification option for batch image copy jobs. The system default value for this parameter is (Y,D). You can specify the following options:

Y Check. If you specify Y, you can specify either of the following parameters:

D If a mismatch is found, deny authorization to use the database.

W Issue a warning message and create a new RDE.

N Do not check.

CHECKRV=
Specifies the verification option for database recovery jobs. The system default value for this parameter is (Y,D). You can specify the following options:

Y Check. If you specify Y, you can specify either of the following parameters:

D If a mismatch is found, deny authorization to use the database.

W Issue a warning message and create a new RDE.

N Do not check.
**CHKRAND=**
Specifies whether to verify changes in randomizing routines by checksum. The system default value for this parameter is N. You can specify the following options:

- **Y**: Check.
- **N**: Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to **Y**, this specification is effective in each IMS environment.

**CHKCOMP=**
Specifies whether to verify changes in segment edit/compression routines by checksum. The system default value for this parameter is N. You can specify the following options:

- **Y**: Check.
- **N**: Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to **Y**, this specification is effective in each IMS environment.

**CHKPSEL=**
Specifies whether to verify changes in HALDB partition selection exit routines by checksum. The system default value for this parameter is N. You can specify the following options:

- **Y**: Check.
- **N**: Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to **Y**, this specification is effective in each IMS environment.

**CHKFPSEL=**
Specifies whether to verify changes in DEDB partition selection exit routines by checksum. The system default value for this parameter is N. You can specify the following options:

- **Y**: Check.
- **N**: Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to **Y**, this specification is effective in each IMS environment.

**GENMAX=**
Specifies the maximum number of expired RDEs to be kept in the LICON data set for use in recoveries. The system default value for this parameter is 0 (do not keep expired RDEs). A maximum of 15 RDE copies can be kept.

**RECUPD=**
Specifies whether to record database update access information. The system default value for this parameter is N. You can specify the following options:

- **Y**: Record.
- **N**: Do not record.
RECLD=
Specifies whether to record database load access information. The system default value for this parameter is N. You can specify the following options:

Y  Record.
N  Do not record.

RECUL=
Specifies whether to record database unload access information. The system default value for this parameter is N. You can specify the following options:

Y  Record.
N  Do not record.

Example

Sample JCL is in the SHPSJCL0 library, member FABLINIT. The following figure shows the CREGOM step of the sample JCL, which creates global option module LIU@INST.

The name of a global option module can be LIU@INST, LIU@imsid, LIUGINST, or LIUGimsid. Replace LIU@INST in the JCL example to create a module with one of these names.

```
//CREGOM EXEC FABLPGEN,MBR=LIU@INST,SOUT=A
//C.SYSLIB DD DISP=SHR,DSN=HPS.SHPSMAC0
//C.SYSIN DD *
LIU@INST FABLPGIN VERIFY=SNGL, single step verification X
 MSGROUT=(2,7,11), WTO message routing codes X
 MSGDESC=7, WTO message descriptor codes X
 VERIFYLMT=5, max number of mismatch messages X
 CHECKON=(Y,D), verification option - online X
 CHECKBAT=(Y,D), verification option - batch X
 CHECKLD=(Y,D), verification option - load X
 CHECKIC=(Y,D), verification option - image copy X
 CHECKRV=(Y,D), verification option - recovery X
 GENMAX=3, max number of expired RDE kept X
 LICON=imshql.licondsn
END
/*
//L.SYSLMOD DD DISP=SHR,DSN=HPS.USERLIB(&MBR)
```

Figure 19. JCL for creating a global option module LIU@INST
The LICON utility provides several functions for handling LICON records.

The functions included are:
- Initialize a LICON data set. A LICON data set must be initialized before it is used.
- Create RDE. You can run a single job to register the DMB information for one, some, or all of the DBD/ACBs in the DBD or ACB library specified.
- Change RDE. You can change an RDE at any time after an RDE is created.
- Delete RDE. If an RDE is no longer needed, you can delete it from the LICON data set.
- Make the current RDE expire, and let the subsequent request for database authorization create a new one.
- Recover a current RDE from an expired RDE.
- List the contents of an RDE. You can print a report of the data stored in an RDE.
- Verify DMB in batch.

JCL requirements for the LICON utility
To run the LICON utility program (FABLIU00), supply an EXEC statement and DD statements.

Subsections:
- “EXEC statement”
- “Summary of DD names” on page 99
- “DD statements” on page 99
- “JCL example” on page 100

EXEC statement

The statement must have the following form:

```
// EXEC PGM=FABLIU00,
//        PARM='IMSID=imsid,IMSPLEX=imsplex,DBRCGRP=dbrcgrp'
```

**IMSID=imsid**

A 4-character IMS ID specifying which global option module is to be used at the IMS subsystem level.

This parameter is optional. If you omit it, the LICON utility determines the IMS ID by use of the batch SCD module (DFSVC000) loaded from the IMS load module library.

**IMSPLEX=imsplex**

A 1 - 5 character IMSplex name to be used for RECON data sets. This parameter is optional.

**DBRCGRP=dbrcgrp**

A 1 - 3 character identifier (ID) assigned to a group of DBRC instances that access the same RECON data set in an IMSplex. This parameter is optional.
Summary of DD names

The following table summarizes the DD names for the LICON utility.

Table 3. DD names for the LICON utility

<table>
<thead>
<tr>
<th>DDNAME</th>
<th>Use</th>
<th>Format</th>
<th>Need</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOBLIB or STEPLIB</td>
<td>Input</td>
<td>PDS</td>
<td>Required</td>
</tr>
<tr>
<td>DFSRESLB</td>
<td>Input</td>
<td>PDS</td>
<td>Required</td>
</tr>
<tr>
<td>FABICON</td>
<td>Input and output</td>
<td>KSIDS</td>
<td>Required (see Note 1)</td>
</tr>
<tr>
<td>FABLPRNT</td>
<td>Output</td>
<td>SYSOUT</td>
<td>Required</td>
</tr>
<tr>
<td>FABLIN</td>
<td>Input</td>
<td>SYSIN</td>
<td>Required</td>
</tr>
<tr>
<td>DBDLIB</td>
<td>Input</td>
<td>PDS</td>
<td>Optional (see Note 2)</td>
</tr>
<tr>
<td>ACBLIB</td>
<td>Input</td>
<td>PDS</td>
<td>Optional (see Note 2)</td>
</tr>
<tr>
<td>RECONx</td>
<td>Input</td>
<td></td>
<td>Optional (see Note 1)</td>
</tr>
</tbody>
</table>

Notes:
1. If dynamic allocation is used, omit the DD statement.
2. Either DBDLIB or ACBLIB DD statement is required.

DD statements

Code the following DD statements to identify the source of input and the placement of output information:

**JOBLIB or STEPLIB DD**

Specifies the load module library, which contains the following resources:
- The LICON utility program (FABLIU00)
- The global option modules (installation level and IMS subsystem level)
- The IMS load module library
- The library that contains DFSMDA dynamic allocation members for the RECON data sets. If the RECONx DD statement is omitted, the library is required (only for HALDBs)
- The library that contains randomizing routines, segment edit/compression exit routines, HALDB partition selection exit, or DEDB partition selection exit when creating a new RDE or verifying a DMB against an RDE with CHKRAND=Y,CHKCOMP=Y,CHKPSEL=Y, or CHKFSEL=Y option. Even if the exit routines are placed in the LPA and LINKLST, you must specify the library that contains them.
- The SCI exit routine for the RECON data sets (optional)

**DFSRESLB DD**

This DD statement is a required DD statement that specifies the library that contains the IMS load modules.

**FABICON DD**

This statement defines the LICON data set. Do not use it if you want to make the LICON utility allocate the LICON data set dynamically.

**DBDLIB DD**

This statement specifies which input DBD library is to be used for the job.

**ACBLIB DD**

This statement specifies which input ACB library is to be used for the job.
Note: Either the DBDLIB or the ACBLIB DD statement is required. If both the DBDLIB and the ACBLIB DD statements are specified in your JCL, the commands of the LICON utility, except the VERIFY.DB command, use the DBDs in the specified DBDLIB data sets.

**FABLPRNT DD**
This statement specifies which output data set contains the report.

**FABLIN DD**
This statement specifies which input control statement stream.

**RECONx DD**
For HALDBs, the RECONx DD statement is required. If this DD statement is omitted, DBRC dynamically allocates the data sets by using DFSMDA dynamic allocation members.

**JCL example**

For a JCL example, see the relevant command topics under "LICON utility reference" on page 98.

**Input for the LICON utility**
The input for the LICON utility must be provided as FABLIN parameters.

**Subsections:**
- "Commands"
- "Control statement syntax"

**Commands**
The following commands are supported:
- INIT.DB
- INIT.LICON
- CHANGE.DB
- DELETE.DB
- EXPIRE.DB
- LIST.DB
- LIST.LICON
- RECOVER.DB
- VERIFY.DB

**Control statement syntax**
The following list describes the coding conventions that you must follow in writing control statements of the LICON utility.
- A control statement can be coded onto one or more lines.
- A control statement cannot contain two or more commands.
- A command and its parameters must be contained between columns 1 - 72.
- A parameter follows a command separated by one or more blanks. When more than one parameter is coded, they must also be separated by one or more blanks.
- A parameter with parentheses ( ) must be coded on the same line.
• A continuation character must be used if a control statement does not fit within a single input record. It is the minus (-) sign.
• Comments consist of character strings beginning with the symbols (/*) and ending with the symbols (*)/. These symbols must be written on the same line.
• Comments can follow the continuation character on each line.

INIT.DB command
INIT.DB command creates an RDE for one, some, or all of the DBDs or ACBs in the specified DBD or ACB library.

Subsections:
• “Syntax”
• “Parameters”
• “IMS environments to be covered” on page 104
• “Response to a mismatch” on page 105
• “Examples” on page 105

Syntax

Parameters

DBD(pattern)
Specifies the database for which you want to create an RDE. You can specify either of the following patterns:
• A specific database name or HALDB partition name
A partially specified database name pattern—for example, DH41*, where *
means ALL.

Note: A partially specified HALDB partition name is not supported. If it is
specified, this command fails with message FABL0451E.

For HALDBs, if you specify a HALDB master name, all the RDEs of its
partitions are created. If you specify a HALDB partition name, only the
RDE of the partition is created.

AREA(pattern)
Specifies the DEDB area for which you want to create an RDE. Only when
you specify a specific DEDB name in the DBD parameter, you can specify
either of the following patterns:

- A specific DEDB area name
- A partially specified DEDB area name pattern

If you omit the keyword, the RDEs for all areas of the DEDB database that
is specified with the DBD parameter are created.

Note for DBD and AREA:
You specify a wildcard in any position in a character string. The asterisk (*)
and the percent sign (%) are supported as wildcard characters. An asterisk
represents 0 - 8 characters, and a percent sign represents a single character.
If two or more asterisks are specified sequentially, only the first asterisk is
recognized.

CHECKON (Y or N, D or W)
Specifies the verification option in effect for online IMS subsystems. You
can specify the following options:

Y  Check. If you specify Y, you can specify either of the following
parameters:
   D  If a mismatch is found, deny authorization to use the
database.
   W  Issue a warning message and create a new RDE.

N  Do not check.

CHECKBAT (Y or N, D or W)
Specifies the verification option in effect for batch jobs. You can specify the
following options:

Y  Check. If you specify Y, you can specify either of the following
parameters:
   D  If a mismatch is found, deny authorization to use the
database.
   W  Issue a warning message and create a new RDE.

N  Do not check.

CHECKLD (Y or N, D or W)
Specifies the verification option in effect for user load program jobs. You
can specify the following options:

Y  Check. If you specify Y, you can specify either of the following
parameters:
If a mismatch is found, deny authorization to use the
database.

Issue a warning message and create a new RDE.

Do not check.

CHECKIC (Y or N, D or W)
Specifies the verification option in effect for batch image copy jobs. You can specify the following options:

Y Check. If you specify Y, you can specify either of the following parameters:

D If a mismatch is found, deny authorization to use the
database.

W Issue a warning message and create a new RDE.

N Do not check.

CHECKRV (Y or N, D or W)
Specifies the verification option in effect for database recovery jobs. You can specify the following options:

Y Check. If you specify Y, you can specify either of the following parameters:

D If a mismatch is found, deny authorization to use the
database.

W Issue a warning message and create a new RDE.

N Do not check.

CHKRAND (Y or N)
Specifies the option to determine whether to verify changes in randomizing routines by checksum. You can specify the following options:

Y Check.

N Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to
Y, this specification is effective in each IMS environment.

CHKCOMP (Y or N)
Specifies the option to determine whether to verify changes in segment edit/compression exit routines by checksum. You can specify the following options:

Y Check.

N Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to
Y, this specification is effective in each IMS environment.

CHKPSEL (Y or N)
Specifies the option to determine whether to verify changes in HALDB partition selection exit routines by checksum. You can specify the following options:

Y Check.

N Do not check.
If CHECKON, CHECKBAC, CHECKLD, CHECKIC, or CHECKRV is set to Y, this specification is effective in each IMS environment.

**CHKFPSEL (Y or N)**

Specifies the option to determine whether to verify changes in DEDB partition selection exit routines by checksum. You can specify the following options:

- **Y** Check.
- **N** Do not check.

If CHECKON, CHECKBAC, CHECKLD, CHECKIC, or CHECKRV is set to Y, this specification is effective in each IMS environment.

**RECUPD (Y or N)**

Specifies the option to determine whether to record database update access information. You can specify the following options:

- **Y** Record.
- **N** Do not record.

**RECLD (Y or N)**

Specifies the option to determine whether to record database load access information. You can specify the following options:

- **Y** Record.
- **N** Do not record.

**RECUL (Y or N)**

Specifies the option to determine whether to record database unload access information. You can specify the following options:

- **Y** Record.
- **N** Do not record.

**GENMAX (nn)**

Specifies the maximum number of expired RDEs to be kept in the LICON data set for use in recoveries. A maximum of 15 RDE copies can be kept.

**REPLACE**

Specifies that an RDE is to be created even if one already exists. The existing RDE expires, and a new one is created. If you do not specify the REPLACE option, and an RDE already exists, the INIT.DB command fails.

**IMS environments to be covered**

Integrity Checker can be invoked in any of five IMS processing environments:

- An online IMS subsystem
- A batch program
- A database loading program
- A batch image copy utility
- A database recovery utility

For each of these environments, you can specify (a) whether DMB verification is to be invoked and (b) whether, if DMB verification is invoked, it is set to issue a nonzero return code for the denial of a request for database authorization, or only to issue a warning message and register the DMB that IMS is using in the LICON data set.
Response to a mismatch

You can choose either of two options as the action to be taken when the DMB that IMS is using for access to the database is not the same as the one registered in the LICON data set:

Deny authorization

Integrity Checker returns a nonzero return code with a reason code of $$ to the requester. It also issues error messages that tell you which value in the DMB is different from the registered one.

Issue a warning and continue

Integrity Checker issues a warning message to notify you of the mismatch, but it continues processing the DMB, and it replaces the RDE in the LICON data set with the one created by the DMB that IMS is using.

Examples

In these examples, the following conditions are assumed:

- The IMS ID is taken from the batch SCD module (DFSVC000) loaded by the IMS load module library IMSVS.SDFSRESL.
- The LICON data set is dynamically allocated by the LICON utility. The data set name is provided by one of the global option modules.

INIT.DB with ACBLIB

This example creates an RDE for every DMB-type ACB member in the ACB library IMSVS.ACBLIB.

```
//LICJOB   JOB
//   EXEC PGM=FABLIU00
//STEPLIB   DD DISP=SHR,DSN=HPS.SHPSLMD0
//   DD DISP=SHR,DSN=IMSVS.SDFSRESL
//ACBLIB    DD DISP=SHR,DSN=IMSVS.ACBLIB
//FABLPRNT  DD SYSOUT=*
//FABLIN    DD *
isons(******)
ifinit.db dbd(*)
/*
```

INIT.DB with DBDLIB

This example creates an RDE for every DBD member in the DBD library IMSVS.DBDLIB.

```
//LICJOB   JOB
//   EXEC PGM=FABLIU00
//STEPLIB   DD DISP=SHR,DSN=HPS.SHPSLMD0
//   DD DISP=SHR,DSN=IMSVS.SDFSRESL
//DBDLIB    DD DISP=SHR,DSN=IMSVS.DBDLIB
//FABLPRNT  DD SYSOUT=*
//FABLIN    DD *
isons(******)
ifinit.db dbd(*)
/*
```
INIT.LICON command

INIT.LICON command initializes the LICON data set so that it can be used.

Subsections:
- “Syntax”
- “Parameters”
- “Example”

Syntax

```plaintext
INIT.LICON
```

Parameters

This command has no parameters.

Example

In this example, the following conditions are assumed:
- The IMS ID is taken from the batch SCD module (DFSVC000) loaded by the IMS load module library IMSVS.SDFSRESL.
- The LICON data set is dynamically allocated by the LICON utility. The data set name is provided by one of the global option modules.

This example initializes the LICON data set for use.

```plaintext
//LICJOB     JOB
  //   EXEC PGM=FABLIU00
//STEPLIB DD DISP=SHR,DSN=HPS.SHPSLMD0
  //   DD DISP=SHR,DSN=IMSVS.SDFSRESL
//FABLPRNT DD SYSOUT=*
//FABLIN DD *
  INIT.LICON
/*
```

CHANGE.DB command

The CHANGE.DB command changes the verification options of the current RDE for the specified non-HALDB full-function database, HALDB partition, or DEDB area.

Subsections:
- “Syntax” on page 107
- “Parameters” on page 107
- “Example” on page 110
Syntax

```
CHANGE.DB DBD(pattern) AREA(pattern)
```

```
CHECKON(Y,D,N) CHECKBAT(Y,D,N)
```

```
CHECKLD(Y,D,N) CHECKIC(Y,D,N)
```

```
CHECKRV(Y,D,N) CHKRAND(Y,N) CHKCOMP(Y,N)
```

```
CHKPSEL(Y,N) CHKFPSEL(Y,N) RECUPD(Y,N)
```

```
RECLD(Y,N) RECU(L(Y,N) GENMAX(nn)
```

Parameters

**DBD(pattern)**

Specifies the database for which you want to change an RDE. You can specify either of the following patterns:

- A specific database name or a specific HALDB partition name
- A partially specified database name pattern or a partially specified HALDB partition name pattern—for example, DH41*, where * means ALL

**Note:** A partially specified HALDB master name is not supported. If it is specified, this command fails with message FABL0461E.

For HALDBs, if you specify a HALDB master name, all the RDEs of its partitions are changed. If you specify a HALDB partition name, only the RDE of the partition is changed.

**AREA(pattern)**

Specifies the DEDB area for which you want to change an RDE. Only when you specify a specific DEDB name in the DBD parameter, you can specify either of the following patterns:

- A specific DEDB area name
- A partially specified DEDB area name pattern

If you omit the keyword, the RDEs for all areas of the DEDB database that is specified with the DBD parameter are changed.
CHECKON (Y or N, D or W)
Specifies the verification option in effect for online IMS subsystems. You can specify the following options:

Y  Check. If you specify Y, you can specify either of the following parameters:
   D  If a mismatch is found, deny authorization to use the database.
   W  Issue a warning message and create a new RDE.

N  Do not check.

CHECKBAT (Y or N, D or W)
Specifies the verification option in effect for batch jobs. You can specify the following options:

Y  Check. If you specify Y, you can specify either of the following parameters:
   D  If a mismatch is found, deny authorization to use the database.
   W  Issue a warning message and create a new RDE.

N  Do not check.

CHECKLD (Y or N, D or W)
Specifies the verification option in effect for user load program jobs. You can specify the following options:

Y  Check. If you specify Y, you can specify either of the following parameters:
   D  If a mismatch is found, deny authorization to use the database.
   W  Issue a warning message and create a new RDE.

N  Do not check.

CHECKIC (Y or N, D or W)
Specifies the verification option in effect for batch image copy jobs. You can specify the following options:

Y  Check. If you specify Y, you can specify either of the following parameters:
   D  If a mismatch is found, deny authorization to use the database.
   W  Issue a warning message and create a new RDE.

N  Do not check.

CHECKRV (Y or N, D or W)
Specifies the verification option in effect for database recovery jobs. You can specify the following options:

Y  Check. If you specify Y, you can specify either of the following parameters:
   D  If a mismatch is found, deny authorization to use the database.
   W  Issue a warning message and create a new RDE.
N  Do not check.

**CHKRAND (Y or N)**
Specifies the option to determine whether to verify changes in randomizing routines by checksum. You can specify the following options:

Y  Check.
N  Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to Y, this specification is effective in each IMS environment.

**CHKCOMP (Y or N)**
Specifies the option to determine whether to verify changes in segment edit/compression exit routines by checksum. You can specify the following options:

Y  Check.
N  Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to Y, this specification is effective in each IMS environment.

**CHKPSEL (Y or N)**
Specifies the option to determine whether to verify changes in HALDB partition selection exit routines by checksum. You can specify the following options:

Y  Check.
N  Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to Y, this specification is effective in each IMS environment.

**CHKFPSEL (Y or N)**
Specifies the option to determine whether to verify changes in DEDB partition selection exit routines by checksum. You can specify the following options:

Y  Check.
N  Do not check.

If CHECKON, CHECKBAT, CHECKLD, CHECKIC, or CHECKRV is set to Y, this specification is effective in each IMS environment.

**RECUPD (Y or N)**
Specifies the option to determine whether to record database update access information. You can specify the following options:

Y  Record.
N  Do not record.

**RECLD (Y or N)**
Specifies the option to determine whether to record database load access information. You can specify the following options:

Y  Record.
N  Do not record.
**RECUL (Y or N)**
Specifies the option to determine whether to record database unload access information. You can specify the following options:

- **Y** Record.
- **N** Do not record.

If RECUPD, RECLD, or RECUL is set to Y and Integrity Checker starts recording database accesses, the recorded information remains until the RDE is re-created or the option is changed from Y to N.

**GENMAX (nn)**
Specifies the maximum number of expired RDEs to be kept in the LICON data set for use in recoveries. A maximum of 15 RDE copies can be kept.

**Example**

In this example, the following conditions are assumed:

- The IMS ID is taken from the batch SCD module (DFSVC000) loaded by the IMS load module library IMSVS.SDFSRESL.
- The LICON data set is dynamically allocated by the LICON utility. The data set name is provided by one of the global option modules.

**CHANGE.DB with DBDLIB**
This example changes the maximum number of expired RDEs to be kept in the LICON data set to 2.

```bash
//LICJOB  JOB
/// EXEC PGM=FABLIU00
//STEPLIB  DD DISP=SHR,DSN=HPS.SHPSLMD0
/// DD DISP=SHR,DSN=IMSVS.SDFSRESL
//DBDLIB  DD DISP=SHR,DSN=IMSVS.DBDLIB
//FABLPRNT DD SYSOUT=*  
//FABLIN   DD *
  CHANGE.DB DBD(DH41TS01) GENMAX(2)
/*
```

**DELETE.DB command**
The DELETE.DB command causes the specified RDEs to be deleted.

**Subsections:**
- “Syntax” on page 111
- “Parameters” on page 111
- “Example” on page 111
Syntax

```
DELETE.DB DBD(pattern)
```

```
AREA(pattern) TIMESTAMP(yyyydddhhmmssth)
```

**Parameters**

**DBD(pattern)**
Specifies the database from which you want to delete an RDE. You can specify either of the following patterns:

- A specific database name or a specific HALDB partition name
- A partially specified database name pattern or a partially specified HALDB partition name pattern—for example, DH41*, where * means ALL

**Note:** A partially specified HALDB master name is not supported. If it is specified, this command fails with message FABL0460E or FABL0461E.

For HALDBs, if you specify a HALDB master name, all the RDEs of its partitions are deleted. If you specify a HALDB partition name, only the RDE of the partition is deleted.

**AREA(pattern)**
Specifies the DEDB area for which you want to delete an RDE. Only when you specify a specific DEDB name in the DBD parameter, you can specify either of the following patterns:

- A specific DEDB area name
- A partially specified DEDB area name pattern

If you omit the keyword, the RDEs for all areas of the DEDB database that is specified with the DBD parameter are deleted.

**TIMESTAMP(yyyydddhhmmssth)**
Specifies the local time stamp value of the RDE you want to delete. You can use the fully specified local time stamp `yyyydddhhmmssth` or the keyword CURRENT for the current RDE. If you omit the keyword, all of the RDEs associated with the DBD are deleted.

**Example**

In this example, the following conditions are assumed:

- The IMS ID is taken from the batch SCD module (DFSVC000) loaded by the IMS load module library IMSVS.SDFSRESL.
- The LICON data set is dynamically allocated by the LICON utility. The data set name is provided by one of the global option modules.

**DELETE.DB with DBDLIB**

This example deletes all RDEs, including the current one and any expired ones, for any database whose name matches the DH41* pattern.
EXPIRE.DB command

The EXPIRE.DB command causes the current RDE for the specified database to expire. An expiry time stamp is assigned to the current RDE.

Subsections:
- “Syntax”
- “Parameters”
- “Example” on page 113

Syntax

```
EXPIRE.DB DBD(pattern) AREA(pattern)
```

Parameters

**DBD(pattern)**

Specifies the database whose current RDE is to expire. You can specify either of the following patterns:

- A specific database name or a specific HALDB partition name
- A partially specified database name pattern or a partially specified HALDB partition name pattern—for example, DH41*, where * means ALL

*Note:* A partially specified HALDB master name is not supported. If it is specified, this command fails with message FABL0461E.

For HALDBs, if you specify a HALDB master name, all the RDEs of its partitions are expired. If you specify a HALDB partition name, only the RDE of the partition is expired.

**AREA(pattern)**

Specifies the DEDB area whose current RDE is to expire. Only when you specify a specific DEDB name in the DBD parameter, you can specify either of the following patterns:

- A specific DEDB area name
- A partially specified DEDB area name pattern

If you omit the keyword, the RDEs for all areas of the DEDB database that is specified with the DBD parameter are expired.
Example

In this example, the following conditions are assumed:

- The IMS ID is taken from the batch SCD module (DFSVC000) loaded by the IMS load module library IMSVS.SDFSRESL.
- The LICON data set is dynamically allocated by the LICON utility. The data set name is provided by one of the global option modules.

**EXPIRE.DB with DBDLIB**

This example expires the current RDE for any database whose name matches the DH41* pattern.

```
//LICJOB JOB
// EXEC PGM=FABLIU00
//STEPLIB DD DISP=SHR,DSN=HPS.SHPSLMD0
// DD DISP=SHR,DSN=IMSVS.SDFSRESL
//DBDLIB DD DISP=SHR,DSN=IMSVS.DBDLIB
//FABLPRT DD SYSOUT=*  
//FABLIN DD *
  EXPIRE.DB DBD(DH41*)
/*
```

**LIST.DB command**

The LIST.DB command shows you the contents of the specified RDEs, with a report.

Subsections:

- “Syntax”
- “Parameters”
- “Example” on page 114

**Syntax**

```
LIST.DB
   DBD(pattern)
       AREA(pattern)
       TIMESTAMP(yyyyddhhmmssth)
       CURRENT
```

**Parameters**

**DBD(pattern)**

Specifies the database for which you want to list the contents of an RDE. You can specify either of the following patterns:

- A specific database name or a specific HALDB partition name
- A partially specified database name pattern or a partially specified HALDB partition name pattern—for example, DH41*, where * means ALL.

**Note:** A partially specified HALDB master name is not supported. If it is specified, this command fails with message FABL0460E or FABL0461E.
For HALDBs, if you specify a HALDB master name, all the RDEs of its partitions are listed. If you specify a HALDB partition name, only the RDE of the partition is listed.

**AREA(pattern)**
Specifies the DEDB area for which you want to list the contents of an RDE. Only when you specify a specific DEDB name in the DBD parameter, you can specify either of the following patterns:
- A specific DEDB area name
- A partially specified DEDB area name pattern

If you omit the keyword, the RDEs for all areas of the DEDB database that is specified with the DBD parameter are listed.

**TIMESTAMP/yyyyydddhhmmssth**
Specifies the local time stamp value of the RDE you want to list. You can use a fully specified local time stamp `yyyyydddhhmmssth` or, to get the current RDE, the keyword CURRENT. If you omit the keyword, all of the RDEs associated with the DBD are listed.

**Example**

In this example, the following conditions are assumed:
- The IMS ID is taken from the batch SCD module (DFSVC000) loaded by the IMS load module library IMSVS.SDFSRESL.
- The LICON data set is dynamically allocated by the LICON utility. The data set name is provided by one of the global option modules.

**LIST.DB with ACBLIB**
This example lists all current RDEs for the database whose name matches the DH41* pattern.

```
//LICJOB   JOB
//  EXEC PGM=FABLIU00
//STEPLIB DD DISP=SHR,DSN=HPS.SHPSLMD0
//ACBLIB DD DISP=SHR,DSN=IMSVS.SDFSRESL
//FABLPRNT DD SYSOUT=* 
//FABLIN DD *
  LIST.DB DBD(DH41*) TIMESTMP(CURRENT)
/*
```

**LIST.LICON command**
The LIST.LICON command evokes a listing of the contents of all the current RDEs in the LICON data set.

**Note:** The report evoked by the LIST.LICON command lists only the current RDEs. It does not contain information about any expired RDE.

**Subsections:**
- “Syntax” on page 115
- “Parameters” on page 115
- “Example” on page 115
Syntax

```
RECOVER.DB DBD(pattern)
   AREA(pattern)
   TIMESTMP(yyyyddhhmmssth)
```

Parameters

There are no parameters for the LIST.LICON command.

Example

In this example, the following conditions are assumed:

- The IMS ID is taken from the batch SCD module (DFSVC000) loaded by the IMS load module library IMSVS.SDFSRESL.
- The LICOND data set is dynamically allocated by the LICOND utility. The data set name is provided by one of the global option modules.

LIST.LICON with ACBLIB

This example lists the current RDEs for all the databases in the LICOND data set.

```
//LICJOB JOB
// EXEC PGM=FABLIU00
//STEPLIB DD DISP=SHR,DSN=HPS.SHPSLMD0
// ACBLIB DD DISP=SHR,DSN=IMSVS.ACBLIB
//FABLPRNT DD SYSPRINT
//FABLIN DD *
LIST.LICON
/*
```

**RECOVER.DB command**

The RECOVER.DB command causes the specified RDE to be recovered by use of the expired RDE specified.

When you issue the RECOVER.DB command, the current RDE expires and the RDE specified by the TIMESTMP parameter becomes current.

Subsections:

- “Syntax”
- “Parameters” on page 116
- “Example” on page 116

**Syntax**

```
RECOVER.DB DBD(pattern)
   AREA(pattern)
   TIMESTMP(yyyyddhhmmssth)
```

Chapter 3. Integrity Checker utility   115
Parameters

DBD(pattern)
Specifies the database from which you want to recover an RDE. You can specify either of the following patterns:

- A specific database name or a specific HALDB partition name
- A partially specified database name pattern or a partially specified HALDB partition name pattern—for example, DH41*, where * means ALL

Note: A partially specified HALDB master name is not supported. If it is specified, this command fails with message FABL0461E.

AREA(pattern)
Specifies the DEDB area for which you want to recover an RDE. Only when you specify a specific DEDB name in the DBD parameter, you can specify either of the following patterns:

- A specific DEDB area name
- A partially specified DEDB area name pattern

If you omit the keyword, the RDEs for all areas of the DEDB database that is specified with the DBD parameter are recovered.

TIMESTMP(yyyyydddhhmmsssth)
Specifies the local time stamp value of the RDE with which you want to recover the current RDE. Specify the entire local time stamp, yyyydddhhmmsssth.

Example

In this example, the following conditions are assumed:

- The IMS ID is taken from the batch SCD module (DFSVC000) loaded by the IMS load module library IMSVS.SDFSRESL.
- The LICON data set is dynamically allocated by the LICON utility. The data set name is provided by one of the global option modules.

RECOVER.DB with ACBLIB
This example changes the expired RDE which has the local time stamp 201506511301302 into a current RDE. At the same time the former current RDE is changed to an expired RDE, in which a time stamp with the time at which this action was taken is set.

```plaintext
//LICJOB  JOB
//     EXEC PGM=FABL1U00
//STEPLIB  DD  DISP=SHR,DSN=HPS.SHPSLMD0
//     DD  DISP=SHR,DSN=IMSVS.SDFSRESL
//ACBLIB  DD  DISP=SHR,DSN=IMSVS.ACBLIB
//FABLPRNT  DD  SYSOUT=*  /*
//FABIN  DD  *
//RECOVER.DB DBD(DH41TS01) TIMESTMP(201506511301302)
/*
```

VERIFY.DB command
The VERIFY.DB command verifies the DBDs or ACBs in the specified DBD or ACB libraries against the RDEs stored in the LICON data set.
Subsections:

- "Syntax"
- "Parameters"
- "Example"

Syntax

```
►►VERIFY.DB—DBD(pattern) AREA(pattern)◄◄
```

Parameters

**DBD(pattern)**

Specifies the database for which you want to verify the DBD/ACBs. You can specify either of the following patterns:

- A specific database name or HALDB partition name
- A partially specified database name pattern. For example, DH41*, where * means ALL

**Note:** A partially specified HALDB partition name is not supported. If it is specified, this command fails with message FABL0451E.

For HALDBs, if you specify a HALDB master name, all the RDEs of its partitions are verified. If you specify a HALDB partition name, only the RDE of the partition is verified.

**AREA(pattern)**

Specifies the DEDB area for which you want to verify the DBD/ACBs. Only when you specify a specific DEDB name in the DBD parameter, you can specify either of the following patterns:

- A specific DEDB area name
- A partially specified DEDB area name pattern

If you omit the keyword, you can verify the DBD/ACBs against the RDEs for all areas of the DEDB database that is specified with the DBD parameter.

You can specify both ACBLIB and DBDLIB DD statements in your JCL.

Example

In this example, the following conditions are assumed:

- The IMS ID is taken from the batch SCD module (DFSVC000) loaded by the IMS load module library IMSVS.SDFSRESL.
- The LICON data set is dynamically allocated by the LICON utility. The data set name is provided by one of the global option modules.

**VERIFY.DB with both DBDLIB and ACBLIB**

This example verifies all current RDEs in the LICON data set.
Output from the LICON utility

Output from the LICON utility consists of the FABLPRNT data set.

FABLPRNT data set

The output from the LICON utility includes the control statements that are specified in the FABLIN data set, the listing of RDE, and the messages issued by Integrity Checker.

The listing of RDE, which is printed only when the LIST.DB command or the LIST.LICON command is specified, contains information about the DMB that is stored in the RDE.

This listing for an RDE record identified by the RDE key contains the following sections:

- Control section
- Historical data section
- If the database access recording option is used, the recorded access information section
- If the database is a HALDB, the PNT section
- If the database is a full-function database, the DMB section.
- If the database is a DEDB, the DMCB section.

The following figure shows an example of the output from the LICON utility.

---

FABL0441I DATABASE: DBDLIUIV TIMESTAMP: CURRENT SUCCESSFULLY PROCESSED
FABL0410I COMMAND COMPLETED WITH RC = 00
FABL0400I LICON UTILITY COMMAND PROCESSING COMPLETE. HIGHEST RC = 00

Figure 20. Output from the LICON utility

Subsections:
Sample report

The following figure shows an example of the listing of RDE.

<table>
<thead>
<tr>
<th>IMS Library Integrity Utilities - LICON Utility</th>
<th>PAGE 0001</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST.OB DBD(*)</td>
<td></td>
</tr>
<tr>
<td>2015.274 20:16:13.23 &quot;LIST OF RDE&quot;</td>
<td>PAGE 0002</td>
</tr>
</tbody>
</table>

RDE Key:

| DBD=DBHDAM10 TIMESTAMP=** CURRENT ** |

Control Section:

| CHECK OPTION FOR ONLINE = (Y,D) |
| CHECK OPTION FOR BATCH = (Y,D)  |
| CHECK OPTION FOR LOAD APL = (Y,D) |
| CHECK OPTION FOR IC = (Y,D)     |
| CHECK OPTION FOR RECOVERY = (Y,D)|
| GENMAX = 1                      |
| CHECK OPTION FOR RANDOMZR = Y    |
| CHECK OPTION FOR COMPRESS = Y    |
| RECORD OPTION FOR UPDATE = Y     |
| RECORD OPTION FOR LOAD = Y       |
| RECORD OPTION FOR UNLOAD = Y     |

Historical Data Section:

| CREATION DATE OF RDE = 2015.274 16:16:12.479195 (LOCAL) |
| LATEST CHANGE DATE OF RDE = N/A                                    |
| RDE FORMAT LEVEL = 2.2                                           |

Recorded Access Information Section:

| LATEST ACCESS FOR UPDATE DATE = 2015.274 17:37:48.404348 (LOCAL) |
| SUBSYSTEM NAME = JOBUPD                                         |
| LATEST ACCESS FOR LOAD DATE = 2015.274 17:30:52.619082 (LOCAL)  |
| SUBSYSTEM NAME = JOBLD                                          |
| LATEST ACCESS FOR UNLOAD DATE = 2015.274 17:31:05.178342 (LOCAL) |
| SUBSYSTEM NAME = JOBUL                                          |

*Figure 21. Listing of RDE (HDAM) (Part 1 of 2)*
Report field descriptions

RDE KEY:

DBD=
DBD name or HALDB partition name

AREA=
AREA name. This field is only for DEDB.

TIMESTAMP=
Time stamp of the RDE. For current RDEs, ** CURRENT ** is shown. For expired RDEs, the expiry date and time are shown in local time, in the format YYYY.DDD HH:MM:SS.TH.

CONTROL SECTION:

CHECK OPTION FOR ONLINE
Check option used for IMS online environment.

CHECK OPTION FOR BATCH
Check option used for IMS batch jobs.

CHECK OPTION FOR LOAD APL
Check option used for IMS database load applications.

CHECK OPTION FOR IC
Check option used for batch image copy jobs.

---

Figure 22. Listing of RDE (HDAM) (Part 2 of 2)

DMB SECTION:

DBD ORGANIZATION = HDAM/VSAM
NUMBER OF DATA SET GROUPS = 2
NUMBER OF SEGMENTS = 3
DATEXIT USED = NO
VERSION ID LENGTH = 13
VERSION ID = 10/01/1519.10

DAC5:

RANDOMIZER NAME = RNM2
HIGHEST RBN IN RAA = 9000
NUMBER OF RAPS PER BLOCK = 3

AMPS:

DSG NUMBER = 1
NUMBER OF RAPS PER BLOCK = 3
PRIME DD NAME = HDAMDD10 OVERFLOW DD NAME = N/A

DSG NUMBER = 2
NUMBER OF RAPS PER BLOCK = 0
PRIME DD NAME = HDAMDD11 OVERFLOW DD NAME = N/A

PSDBS:

SEGMENT CODE = 01 PARENT SEGMENT CODE = 00
HIERARCHICAL LEVEL = 1
PTR NUM IN PARENT TO 1ST = 0 PTR NUM IN PARENT TO LAST = 0
PREFIX SIZE = 18 PREFIX FLAGS = 60
DATA LENGTH = 28
INSERT RULES = 23 DELETE/REPLACE RULES = 45

...
CHECK OPTION FOR RECOVERY
Check option used for IMS database recovery jobs.

GENMAX
The number of expired RDEs for the database to be kept.

CHECK OPTION FOR RANDOMZR
Check option used for a randomizing routine.

CHECK OPTION FOR COMPRESS
Check option used for segment edit/compression exit routines.

CHECK OPTION FOR PART SEL
Check option used for a HALDB partition selection exit routine.

CHECK OPTION FOR FP PSEL
Check option used for a DEDB partition selection exit routine.

RECORD OPTION FOR UPDATE
Record option used for recording database update accesses.

RECORD OPTION FOR LOAD
Record option used for recording database load accesses (except in IMS online environments).

RECORD OPTION FOR UNLOAD
Record option used for recording database unload accesses.

HISTORICAL DATA SECTION

CREATION DATE OF RDE
The creation date of the RDE is shown in local time and in UTC, in the format YYYY.DDD HH:MM:SS.THMIJU. If the RDE has been converted by the LICON data set migration utility, UTC is shown as N/A.

LATEST CHANGE DATE OF RDE
The latest change date of the RDE is shown in local time and in UTC, in the format YYYY.DDD HH:MM:SS.THMIJU. If the RDE has not been changed since creation, N/A is shown. If the RDE has been converted by the LICON data set migration utility, UTC is shown as N/A.

RDE FORMAT LEVEL
The level of the RDE record format.

RECOVERABLE WITH TIMESTAMP
Information that indicates whether Integrity Checker verifies if the RDE was valid for the DMB verification, at the specific recovery time—the point in time to which a non-HALDB full-function database, a HALDB partition, or a DEDB area is to be recovered by using time stamp recovery of IMS HP Image Copy.

Y  Integrity Checker verifies whether the RDE was valid at the specific recovery time. If Integrity Checker determines that it was valid at that time, Integrity Checker recovers the RDE and uses it for the DMB verification.

N  Integrity Checker does not verify whether the RDE was valid at the specific recovery time.

RECORDED ACCESS SECTION
This section contains recorded database access information for update, load, and unload operations.
LATEST ACCESS FOR UPDATE
This field shows the information about the latest database access made for update operation. This field is shown when the record option for database update access is specified as Y.

DATE The date and the time of the latest access for database update operation is shown in local time and in UTC, in the format YYYY:DDD HH:MM:SS.THMIJU. When database update access information is not recorded, this field shows N/A.

SUBSYSTEM The name of the subsystem that most recently accessed the databases for database update operation. The subsystem name is the same as the name recorded in the RECON data set. When database update access information is not recorded, this field shows N/A.

LATEST ACCESS FOR LOAD
This field shows the information about the latest database access made for load operation. This field is shown when the record option for database load access is specified as Y.

DATE The date and the time of the latest access for database load operation is shown in local time and in UTC, in the format YYYY:DDD HH:MM:SS.THMIJU. When database load access information is not recorded, this field shows N/A.

SUBSYSTEM The name of the subsystem that most recently accessed the databases for database load operation. The subsystem name is the same as the name recorded in the RECON data set. When database load access information is not recorded, this field shows N/A.

LATEST ACCESS FOR UNLOAD
This field shows the information about the latest database access made for unload operation. This field is shown when the record option for database unload access is specified as Y.

DATE The date and the time of the latest access for database unload operation is shown in local time and in UTC, in the format YYYY:DDD HH:MM:SS.THMIJU. When database unload access information is not recorded, this field shows N/A.

SUBSYSTEM The name of the subsystem that most recently accessed the databases for database unload operation. The subsystem name is the same as the name recorded in the RECON data set. When database unload access information is not recorded, this field shows N/A.

PNT SECTION (for HALDBs)
This section contains information about the PNT. It also contains information related to the HALDB partition.

DMB SECTION (for full-function databases)
This section contains the information about the DMB. It also contains the following subsections:
DACS  Information related to randomizing parameters. This field is only for HDAM.

AMPS  Information related to the data set groups.

PSDBS  Information related to the segments.

DMCB SECTION (for DEDBs)
This section contains the information about the DMCB. It also contains the following subsections:

SDBF  Information related to the segments.

DMAC  Information related to the area.

CRTE  Information related to the indexes.

Each item has a self-explanatory label to help you understand the information.
Chapter 4. Consistency Checker utility

The Consistency Checker utility helps you ensure that the necessary definitions in an IMS subsystem have been created for your database or your application program.

Topics:

- “Consistency Checker utility overview” on page 126
- “Restriction for Consistency Checker” on page 129
- “Checking the consistency of definitions” on page 130
- “JCL requirements for the Consistency Checker utility” on page 131
- “Control statements for the Consistency Checker utility” on page 134
- “JCL examples for the Consistency Checker utility” on page 137
- “Output from the Consistency Checker utility” on page 140
Consistency Checker utility overview

The Consistency Checker utility ensures that the definitions necessary to an IMS subsystem have been created for your database or your application program.

Subsections:
- “Function overview”
- “Program structure” on page 127
- “Data flow” on page 127

Function overview

For a DBD in the DBD library, Consistency Checker verifies whether the following definitions have been created correctly in each library and whether these definitions are consistent with the DBD:
- The ACB in the ACB library
- The database definition entry in the MODBLKS module
- The database definition entry in the resource definition data sets (RDDSs)
- The DFSMDA dynamic allocation member for the database data set in the MDA library
- The DB and DSG registration record in the RECON

Consistency Checker determines which type of library is to verify depending on both the user input and the database or organization defined in the specified DBD, as shown in the following table.

<table>
<thead>
<tr>
<th>Database organization</th>
<th>ACBLIB (ACBLIB data set is specified)</th>
<th>MODBLKS (MODBLKS data set is specified and DRD=NO is specified)</th>
<th>DFSMDA (DFSMDA data set is specified)</th>
<th>RECON (CHKRECON=YES is specified)</th>
<th>DRD (DRD=YES is specified)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSAM (including SHSAM)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HISAM (including SHISAM)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HDAM</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HIDAM</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INDEX (including Fast Path secondary indexes)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PHDAM</td>
<td>Yes</td>
<td>Yes</td>
<td>No (see note 1)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PHIDAM</td>
<td>Yes</td>
<td>Yes (see note 2)</td>
<td>No (see note 1)</td>
<td>Yes</td>
<td>Yes (see note 2)</td>
</tr>
<tr>
<td>PSINDEX</td>
<td>Yes</td>
<td>Yes (see note 2)</td>
<td>No (see note 1)</td>
<td>Yes</td>
<td>Yes (see note 2)</td>
</tr>
<tr>
<td>GSAM</td>
<td>No (see note 3)</td>
<td>No (see note 4)</td>
<td>No (see note 3)</td>
<td>No (see note 5)</td>
<td>No (see note 4)</td>
</tr>
<tr>
<td>LOGICAL</td>
<td>No (see note 3)</td>
<td>No (see note 4)</td>
<td>No (see note 6)</td>
<td>No (see note 6)</td>
<td>No (see note 4)</td>
</tr>
<tr>
<td>MSDB</td>
<td>Yes (see note 6)</td>
<td>Yes</td>
<td>No (see note 6)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>DEDB</td>
<td>Yes</td>
<td>Yes</td>
<td>No (see note 7)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Notes:
1. PHDAM and PHIDAM are verified only if the database is a Fast Path database.
2. PSINDEX is verified only if the database is a Fast Path database.
3. GSAM is verified only if the database is a Fast Path database.
4. LOGICAL is verified only if the database is a Fast Path database.
5. MSDB is verified only if the database is a Fast Path database.
6. DEDB is verified only if the database is a Fast Path database.
7. Tables and figures are not supported in this mode.
Table 4. Verified libraries for each database organization (continued)

<table>
<thead>
<tr>
<th>Database organization</th>
<th>ACBLIB (ACBLIB data set is specified)</th>
<th>MODBLKS (MODBLKS data set is specified and DRD=NO is specified)</th>
<th>DFSMDA (DFSMDA data set is specified)</th>
<th>RECON (CHKRECON=YES is specified)</th>
<th>DRD (DRD=YES is specified)</th>
</tr>
</thead>
</table>

Notes:
1. The DFSMDA dynalloc allocation member for HALDB is not used by IMS.
2. There is no need to create a MODBLKS or RDDS entry for IMS catalog databases.
3. No ACB member is generated for GSAM or LOGICAL databases.
4. There is no need to create a MODBLKS entry or an RDDS entry for GSAM or LOGICAL databases.
5. There is no need to register GSAM databases to RECON.
6. No database data set is generated for LOGICAL or MSDB databases.
7. For a DEDB database, there is no need to supply DFSMDA member if the database data sets are registered to RECON. Consistency Checker does not verify DFSMDA members for DEDB area data sets.

For a PSB in the PSB library, Consistency Checker verifies whether the following definitions have been created correctly in each library and whether these definitions are consistent with the PSB:

- The ACB in the ACB library
- The application program definition entry in the MODBLKS module
- The application program definition entry in the resource definition data sets (RDDSs)

Notes:
- Consistency Checker does not verify GSAM PCBs because no ACB member is generated.
- Consistency Checker does not verify the MODBLKS or RDDS entry when DBDs or PSBs of IMS catalog databases are processed because these entries are not required.

Consistency Checker generates a DBD Check report, a PSB Check report, or both, after each check and helps you determine which definitions are needed before you start an IMS subsystem.

Program structure

Consistency Checker is provided as an MVS batch utility program, and depending on the specifications on the control statements, its functions can be invoked from the load module FABLECHK.

Data flow

The following figure shows the general data flow for Consistency Checker (FABLECHK). The input consists of the SYSIN data set (contains the control statements), the DBDLIB data set, the PSBLIB data set, the ACBLIB data set, the DFSMDA data set, the MODBLKS data set, the resource definition data set, and RECON data sets. The output consists of reports and an activity log.
Figure 23. Data flow for Consistency Checker
Restriction for Consistency Checker

Certain restriction applies when you use the Consistency Checker utility.

Consistency Checker does not support verification of IMS catalog partition definition data sets. If CHKRECON=YES is specified and if the IMS catalog database is not registered to the RECON data set, Consistency Checker determines that the definition is not consistent with the DBDs of the IMS catalog database.
Checking the consistency of definitions

To check the consistency of definitions that are required to run the IMS subsystem by using the Consistency Checker utility, you must prepare the JCL for the Consistency Checker utility, submit the job, and check the DBD and PSB Check reports.

About this task

Sample JCL for the Consistency Checker utility is in the SHPSJCL0 library, member FABLIVP2. You can modify this sample JCL and then use it to run the utility.

Procedure

1. In the Consistency Checker JCL, code the EXEC statement and DD statements. See “JCL requirements for the Consistency Checker utility” on page 131.
2. In the SYSIN data set, code the control statements for Consistency Checker. See “Control statements for the Consistency Checker utility” on page 134.
3. Submit the job.
4. Check the output data sets that are generated. See “Output from the Consistency Checker utility” on page 140.
JCL requirements for the Consistency Checker utility

When you code the JCL to run the Consistency Checker utility, include the EXEC statement and appropriate DD statements.

Subsections:
- "JCL example"
- "EXEC statement"
- "DD statements"

JCL example

The following figure shows the JCL that is required for checking consistency of DBDs and PSBs.

```
//stepname EXEC PGM=FABLECHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
  DD DSN=IMSVS.DFSMDA,DISP=SHR
//DFSRESLB DD DSN=IMSVS.SDFSRESL,DISP=SHR
//DBDLIB DD DSN=IMVS.DBDLIB,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//DFSMDA DD DSN=IMSVS.DFSMDA,DISP=SHR
//MODBLKS DD DSN=IMSVS.MODBLKS,DISP=SHR
//SYSRDDS DD DSN=IMSVS.SYSRDDS,DISP=SHR
//NSYSRDDS DD DSN=IMSVS.NSYSRDDS,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
  (control statements)
/*
```

Figure 24. JCL for checking consistency of DBDs and PSBs

EXEC statement

The EXEC statement must be in the following format:
```
//stepname EXEC PGM=FABLECHK,PARM='IMSPLEX=imsplex, DBRCGRP=dbrcgrp'
```

IMSPLEX=imsplex

A 1 - 5 character IMSplex name used for RECON data sets. This parameter is optional.

DBRCGRP=dbrcgrp

A 1 - 3 character identifier (ID) assigned to a group of DBRC instances that access the same RECON data set in an IMSplex. This parameter is optional.

DD statements

Code the following DD statements to identify the source of input and the placement of output information.

Consistency Checker verifies the libraries that the input DD statements specify, however, depending on the type of database organization, Consistency Checker does not verify certain libraries. For more information, see [Verified libraries for each database organization](#).

Chapter 4. Consistency Checker utility  131
STEPLIB DD
Required. This DD statement defines the input data sets as follows:
- IMS Library Integrity Utilities load module library (required)
- The library that contains DFSMDA dynamic allocation members for the RECON data set. When the RECON DD statement is omitted, this DD is required. When the members are included in the library that the DFSMDA DD statement specifies, this DD is not required.
- The library that contains the IMS load modules, and is optional. When CHKRECON=YES is specified, this DD statement is required.
- If you use the SCI exit routine for your IMS environment, specify the load module data set that contains the exit routine.

DFSRESLB DD
Required. This input DD statement points to the library that contains the IMS load modules.

DBDLIB DD
Required if the DBD control statement is specified. This input DD statement points to the library that contains the DBDs to check.

PSBLIB DD
Required if the PSB control statement is specified. This input DD statement points to the library that contains the PSBs to check.

ACBLIB DD
Optional. This input DD statement points to the library that contains the ACBs to check. If you specify this DD statement, Consistency Checker verifies whether the ACB member that corresponds to the specified DBD or PSB has been created. If the ACB member exists, Consistency Checker verifies whether the member is consistent with the DBD or the PSB. If this data set is not specified, Consistency Checker does not check ACBs for any DBD or PSB.

DFSMDA DD
Optional. This input DD statement points to the library that contains the DFSMDA dynamic allocation members. If you specify this DD statement, Consistency Checker verifies whether the DFSMDA dynamic allocation member that corresponds to the specified DBD has been created. If the dynamic allocation member exists, Consistency Checker verifies whether the DSG registration record is consistent with the DBD. If this data set is not specified, Consistency Checker does not check DFSMDA members for any DBDs.

MODBLKS DD
Optional. This input DD statement points to the library that contains MODBLKS, which are control block modules created by IMS system definition. If the data set is specified and DRD=NO is specified in the SYSIN data set, Consistency Checker verifies whether an entry that corresponds to the specified DBD or PSB has been created in the MODBLKS module and, if so, whether it is consistent with the DBD or the PSB. If this data set is not specified, Consistency Checker does not check the MODBLKS for any DBD or PSB.

SYSRDDS DD
Optional. This input DD statement points to the data set that contains the system RDDs, which are defined in DRD environments. If the data set is specified and DRD=YES is specified in the SYSIN data set, Consistency Checker verifies whether a definition that corresponds to the specified
DBD or PSB has been created in the RDDS and, if created, whether it is consistent with the DBD or the PSB. If this data set is not specified, Consistency Checker does not check the RDDS for any DBD or PSB. Consistency Checker verifies only the latest RDDS even if multiple RDDSs are specified.

**NSYSRDDS DD**
Optional. This input DD statement points to the data set that contains the non-system RDDS, which is not defined in DRD environments. If the data set is specified and DRD=YES is specified in the SYSIN data set, Consistency Checker verifies whether a definition that corresponds to the specified DBD or PSB has been created in the RDDS and, if created, whether it is consistent with the DBD or the PSB. If this data set is not specified, Consistency Checker does not check the RDDS for any DBD or PSB. The data set must be created by the EXPORT TYPE(ALL) NAME(*) command to contain all the resource definitions. Concatenated data sets are not used.

**RECONx DD**
Optional. These input DD statements point to the RECON data sets. If the data set is specified and CHKRECON=YES is specified in the SYSIN data set, Consistency Checker verifies whether a definition that corresponds to the specified DBD has been created in the RECON data set. If the definition exists, Consistency Checker verifies whether the definition and the DBDS records in the RECON data sets are consistent with the DBD. If this DD statement is omitted, DBRC dynamically allocates the data sets by using DFSMDA dynamic allocation members when Consistency Checker issues a DBRC command internally.

**SYSOUT DD**
Required. This output DD statement points the data set for generating all activity logs and error messages. The record format is fixed-blocked. The logical record length is 133. Block size, if coded, must be a multiple of 133.

**SYSPRINT DD**
Required. This output DD statement points to the data set in which Consistency Checker generates the DBD check report, the PSB check report, or both. Each report is sorted alphabetically by member name. The record format is fixed-blocked. The logical record length is 133. Block size, if coded, must be a multiple of 133.

**SYSIN DD**
Required. The SYSIN DD is the control data set for this program. The record format is fixed-blocked. The logical record length is 80. Block size, if coded, must be a multiple of 80. You can specify up to 9999 control statements by using the SYSIN DD statement.

**Related reading:** See “Control statements for the Consistency Checker utility” on page 134 for control statements.
Control statements for the Consistency Checker utility

The input to the Consistency Checker utility consists of control statements in the SYSIN data set.

Subsections:
- “Control statement example”
- “Syntax rules”
- “Control statement keywords”

Control statement example

The following figure shows the control statements that can be coded in the SYSIN data set.

```
//SYSIN DD *
  DDIRSFX='A'
  CHKRECON=YES
  FAILONLY=YES
  FAILRC=04
  DBD=TESTDB1
  DBD=TESTDB2
/*
```

Figure 25. Example of the control statements for Consistency Checker

Syntax rules

The following guidelines apply to the control statements for Consistency Checker:

- The control statements can be coded anywhere between columns 2 - 80.
- In the control statement field, blanks must not be used between keyword, equal sign, and member name.
- Comments can be written after a blank because a blank is considered the delimiter.
- Statements with an asterisk (*) in column 1 are treated as comments.

Control statement keywords

The format of each control statement is as follows:

Note: DDIRSFX, PDIRSFX, CHKRECON, PCBERRLMT, FAILONLY, and FAILRC statements can be specified only once, in no special order. DDIRSFX and CHKRECON are effective for all DBD members, PDIRSFX and PCBERRLMT are effective for all PCB members, and FAILONLY and FAILRC are effective for all DBD and PSB members.

**DDIRSFX=[x10]**

This statement specifies the alphanumeric suffix character appended to DFSDDIR of the MODBLKS module name. The default is 0. If MODBLKS data set is specified in the MODBLKS DD statement, this specification is effective. If DRD=YES is specified in the SYSIN data set, this specification is ignored.

**PDIRSFX=[x10]**

This statement specifies the alphanumeric suffix character that is appended
to DFSPDIR of the MODBLKS module name. The default is 0. If MODBLKS data set is specified in the MODBLKS DD statement, this specification is effective. If DRD=YES is specified in the SYSIN data set, this specification is ignored.

**CHKRECON=YES | NO**
This statement specifies whether to verify the registration of the database and the data set to RECON. The default is NO.

**FAILONLY=YES | NO**
This statement specifies whether the DBD check reports are to be printed only for the DBDs that fail the consistency check, and whether the PSB check reports are to be printed only for the PSBs and the PCBs that fail the consistency check. The default is NO, which means these check reports are printed for all of the DBDs and for all of the PSBs to be checked.

**FAILRC=nn**
This statement specifies the return code by two-digit decimal number which is returned when the consistency check fails for any DBDs or any PSBs. The default is 08.

**PCBERRLMT=nnnn**
This statement specifies the maximum number of inconsistent PCBs in each PSB that is to be printed on the PSB check report. If the number of inconsistent PCBs in each PSB has exceeded the value specified in this statement, Consistency Checker will not check further PCBs. You can specify a left-aligned decimal number in the range of 0 - 2500. This specification is effective only when FAILONLY=YES is specified. The default is 2500.

**NOCHKORG=dborg**
This statement specifies that the database organization is to be excluded from the consistency check. One or more of the following types can be specified:
- HSAM, HISAM, HDAM, HIDAM, INDEX, PHDAM, PHIDAM, PSINDEX, GSAM, LOGICAL, MSDB, and DEDB.

**Note:** Here, HSAM includes SHSAM, and HISAM includes SHISAM.

For example, specifying the following statement causes the DBDs that are defined as HSAM, SHSAM, or GSAM to be excluded from consistency checking.

```plaintext
NOCHKORG=HSAM,GSAM
```

When specifying two or more statements, specify them as follows:

```plaintext
NOCHKORG=HSAM
NOCHKORG=GSAM
```

**DRD=YES | NO**
This statement specifies whether Dynamic Resource Definition (DRD) is enabled in your IMS system. The default is NO. If an RDDS data set is specified in the SYSRDDS or NSYSRDDS DD statement, this specification is effective.

**DBD=member**
This statement specifies the DBD member names to be checked. You can use wildcards to specify multiple members.
**PSB=member**

This statement specifies the PSB member names to be checked. You can use wildcards to specify multiple members.

**Note:** For the DBD and PSB control statements, wildcards that can be used are an asterisk (*) and a percent sign (%). An asterisk represents 0 - 8 characters, and a percent sign represents a single character. If two or more asterisks are specified sequentially, only the first asterisk is recognized.
JCL examples for the Consistency Checker utility

The following topics provide JCL examples for running the Consistency Checker utility to check the consistency of DBDs and PSBs.

Example: Checking the consistency of DBDs

The following figure shows example JCL for checking whether the ACB member, the MODBLKS module, the DFSMDA member, and the RECON data are consistent with each DBD in the specified DBD library.

In this example, the RECON data sets are allocated dynamically by use of the DFSMDA dynamic allocation members in the library that is specified in the DFSMDA DD statement, without the specification of the RECONn DD statement.

In the SYSIN data set, in addition to the DBD member names to be checked, the following optional control statements are specified:

- DDIRSFX=A specifies that the MODBLKS module name to be verified is DFSDDIRA.
- CHKRECON=YES specifies to verify the registration in the RECON data set.
- FAILONLY=YES specifies that the DBD check reports are printed only for DBDs whose consistency check fails.
- FAILRC=08 specifies the return code is 08 when the consistency check fails for any DBDs.
- DBD=* specifies that all DBDs in the specified DBD library are to be checked.

```jcl
//stepname EXEC PGM=FABLECHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMSVS.SDFSRESL,DISP=SHR
//DFSRESLB DD DSN=IMSVS.SDFSRESL,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//MODBLKS DD DSN=IMSVS.MODBLKS,DISP=SHR
//DFSMDA DD DSN=IMSVS.DFSMDA,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
 DDIRSFX=A
 CHKRECON=YES
 FAILONLY=YES
 FAILRC=08
 DBD=*```

Figure 26. Checking the consistency of DBDs
Example: Checking the consistency of PSBs

The following figure shows example JCL for checking whether the ACB member and the MODBLKS module are consistent with each PSB in the specified PSB library.

In the SYSIN data set, in addition to the PSB member names to be checked, the following optional control statements are specified:

- `PDIRSFX=A` specifies that the MODBLKS module name to be verified is DFSPDIRA.
- `FAILONLY=YES` specifies that the PSB check reports are printed only for PSBs and PCBs whose consistency check fails.
- `PCBERRLMT=5` specifies the maximum number of the inconsistent PCBs in each PSB that is to be printed on the PSB check report.
- `PSB=*` specifies that all PSBs in the specified PSB library are to be checked.

```
//stepname EXEC PGM=FABLECHK
//STPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DFSRESLB DD DSN=IMSVS.SDFSRESL,DISP=SHR
// PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
// ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
// MODBLKS DD DSN=IMSVS.MODBLKS,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *

  PDIRSFX=A
  FAILONLY=YES
  PCBERRLMT=5
  PSB=*  
/*
```

Figure 27. Checking the consistency of PSBs
Example: Checking the consistency of DBDs and PSBs

The following figure shows example JCL for checking whether the ACB member and the MODBLKS module are consistent with each DBD and PSB in the specified DBD and PSB library.

```jcl
//stepname EXEC PGM=FABLECHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DFSRESLB DD DSN=IMSVS.SDFSRESL,DISP=SHR
// DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
// PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
// ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
// MODBLKS DD DSN=IMSVS.MODBLKS,DISP=SHR
// SYSCOUT DD SYSOUT=A
// SYSPRINT DD SYSOUT=A
//SYSIN DD *
  DBD=TESTDB1
  DBD=TESTDB2
  PSB=TESTPSB1
  PSB=TESTPSB2
/*
```

Figure 28. Checking the consistency of DBDs and PSBs
Output from the Consistency Checker utility

Consistency Checker generates the outputs in the SYSOUT data set and the SYSPRINT data set.

SYSOUT data set

The SYSOUT data set contains activity logs and error messages issued by Consistency Checker.

The messages also show the following information:
- Which parameter is applied to each control statement
- Whether the specified DBD was found in the DBD library
- Whether the consistency check for each DBD succeeded or failed
- Whether the specified PSB was found in the PSB library
- Whether the consistency check for each PSB succeeded or failed

The following figure shows messages that are generated in the SYSOUT data set.

SYSPRINT data set

The SYSPRINT data set contains the DBD Check reports, the PSB Check reports, or both. Each report is sorted alphabetically by member name.

The SYSPRINT data set must contain fixed-length records of 133 bytes, and a block size of 133 or a multiple of 133.
**DBD Check report**

The DBD Check report is generated in the SYSPRINT data set as a result of the consistency check for DBDs.

This report contains the following three parts:
- Library information
- Database (DB) information
- Data set group (DSG) information

The library information part is printed once, on the first page. If FAILONLY=YES statement is specified in SYSIN DD, the database information part and the data set group information part are printed only for the DBDs for which any inconsistency was detected in the consistency check. Otherwise, these parts are printed for all DBDs.

**Subsections:**
- “Sample report”
- “Report field descriptions for the library information part” on page 142
- “Report field descriptions for the DB information part” on page 143
- “Report field descriptions for the DSG information part” on page 146

**Sample report**

The following figure shows an example of the DBD Check report.

---

**Figure 30. DBD Check report (Part 1 of 2)**
Report field descriptions for the library information part

This part contains information about the input libraries for the DBD check.

**IMS**
IMS version and release number retrieved from the library specified in DFSRESLB DD

**RECON**
RECON version and release number retrieved by the DBRC command

The following lines show the volume name and the data set name of each library.

**DFSRESLB**
IMS load library specified in DFSRESLB DD

**DBDLIB**
DBD library specified in DBDLIB DD
ACLIB
ACB library specified in ACLIB DD
Unless each data set is specified in the DD statement, the line for the library is not shown.

MODBLKS
MODBLKS module library specified in MODBLKS DD
MODBLKS information is not printed in the following cases:
• When DRD=YES is specified in SYSIN DD.
• When the data set is not specified on the DD statement.
• When an IMS catalog DBD is processed.

SUFFIX = (x)
Suffix appended to DFSDDIR of the MODBLKS module name

SYSRDDS
System RDDS specified in SYSRDDS DD
SYSRDDS information is not printed in the following cases:
• When DRD=NO is specified in SYSIN DD.
• When the data set is not specified on the DD statement.
• When an IMS catalog DBD is processed.

NSYSRDDS
Non-system RDDS specified in NSYSRDDS DD
NSYSRDDS information is not printed in the following cases:
• When DRD=NO is specified in SYSIN DD.
• When the data set is not specified on the DD statement.
• When an IMS catalog DBD is processed.

DFSMDA
DFSMDA dynamic allocation module library specified in DFSMDA DD
Unless each data set is specified in the DD statement, the line for the library is not shown.

RECONn
DBRC RECON data set name
If CHKRECON=NO is specified in SYSIN DD, the lines for RECONn are not shown.

Report field descriptions for the DB information part

This part contains information about each database definition and the results of the consistency check.

For each DBD, Consistency Checker decides which type of library is to be verified depending on both the user input and the database organization defined in the DBD as shown in Table 4 on page 126. In this part, information about the only libraries to be verified is printed.

DBD NAME
DBD name. Volume serial number and data set name of the library that contains the DBD member.

The columns of the table in the DB information part are as follows:
LIBRARY
  Library that contains each definition

CHK  The mark *** is shown if any inconsistency is detected between the DBD and each definition

ITEM/FIELD
  Definition item or field

CONTENTS
  Contents of each field

DBDLIB
  Contents of the DBDs field

The rows of the table in the DB information part are as follows:

<table>
<thead>
<tr>
<th>ACBLIB</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACB MBR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GENDATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>The date and time when the ACB was generated.</td>
</tr>
</tbody>
</table>

  **Note:** If the generation date of the DBD is later than the generation date of the ACB, Consistency Checker regards them as inconsistent.

<table>
<thead>
<tr>
<th>IMSREL</th>
</tr>
</thead>
<tbody>
<tr>
<td>The IMS version and release that generated the ACB.</td>
</tr>
</tbody>
</table>

  **Note:** If they are different from the IMS version and release defined in the DFSRESLIB library, Consistency Checker regards them as inconsistent.

<table>
<thead>
<tr>
<th>ACCESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>The DL/I access method and the operating system access method.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SEGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>The number of segments in the database. If it is a DEDB database, the number does not include the dummy segments.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RMNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>The name of randomizing module. This field is shown only for an HDAM, a PHDAM, or a DEDB database.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ANCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>The number of root anchor points in each control interval or block. This field is shown only for an HDAM or a PHDAM database.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RBN</th>
</tr>
</thead>
<tbody>
<tr>
<td>The maximum relative block number value. This field is shown only for an HDAM or a PHDAM database.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BYTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>The maximum number of bytes of database record that can be stored in the root addressable area (RAA). This field is shown only for an HDAM or a PHDAM database.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>XCI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whether this DEDB uses the Extended Call Interface when making calls to the randomizer or not. This field is shown only for a DEDB database.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MODBLKS</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>144</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS Library Integrity Utilities User's Guide</td>
</tr>
</tbody>
</table>
DB DEF
Whether a database definition that corresponds to the DBD is found in the specified MODBLKS module.

ACCSDLVL
The access for the defined database:
EX: Exclusive
UP: Update
RO: Read only
RD: Read

MODBLKS information is not printed in the following cases:
• When DRD=YES is specified in SYSIN DD.
• When the data set is not specified on the DD statement.
• When an IMS catalog DBD is processed.

SYSRDDS
DB DEF
Whether a database definition that corresponds to the DBD is found in the specified system RDDS.

ACCSDLVL
The access for the defined database:
EX: Exclusive
UP: Update
RO: Read only
RD: Read

SYSRDDS information is not printed in the following cases:
• When DRD=NO is specified in SYSIN DD.
• When the data set is not specified on the DD statement.
• When an IMS catalog DBD is processed.

NSYSRDDS
DB DEF
Whether a database definition that corresponds to the DBD is found in the specified non-system RDDS.

ACCSDLVL
The access for the defined database:
EX: Exclusive
UP: Update
RO: Read only
RD: Read

NSYSRDDS information is not printed in the following cases:
• When DRD=NO is specified in SYSIN DD.
• When the data set is not specified on the DD statement.
• When an IMS catalog DBD is processed.

DFSMDA
MDA MBR
Whether a DFSMDA dynamic allocation member that corresponds to the DBD is found in the DFSMDA data set.

RECON
**DB RECORD**
Whether a DB record that corresponds to the DBD is found in the RECON.

**SHRLVL**
The level of data sharing (0, 1, 2, or 3).

**TYPE**
The type of database (FP, IMS, or HALDB).

**DBORG/DSORG**
The database organization and the data set organization.

**PARTITIONS**
The number of partitions. This field is shown only for PHDAM, PHIDAM, or PSINDEX databases.

**BACKOT NEED**
The BACKOUT NEEDED flag (ON or OFF). For DEDB, this field is not shown.

**PROHBT AUTH**
The PROHIBIT AUTHORIZATION flag (ON or OFF).

**RECOV NEED#**
The count of RECOVERY NEEDED.

**IC NEED#**
The count of IMAGE COPY NEEDED.

**OLR CAPABLE**
The online reorganization capable flag (YES or NO). This field is shown only for PHDAM, PHIDAM, or PSINDEX databases.

---

**Report field descriptions for the DSG information part**

This part contains information about data set group definitions and results of the consistency check.

The columns of the table in the DSG information part are as follows:

**DSG #**
The sequential number of the data set groups.

**CHK**
The mark *** is shown if any inconsistency is detected between each definition and the DBD. Additionally the mark * is shown on the right side of each data which is determined as inconsistent.

**DD**
This column indicates one of the following DD of the data set group:
1: DD1
2: DD2 or OVFLW

**ACBLIB, DBDLIB**
These columns show the following fields found in the DBD and the ACB:

**DD1/DD2**
The ddname of the data set. For a HALDB, the ddname is retrieved from the specification on the DSGROUP parameter of the SEGMENT statement in the DBD.

**BLKSIZE**
The block size of the data set. The value is shown only for (HDAM,OSAM) or (HIDAM,OSAM). For other databases, N/A is shown.
MR Identifies whether the information on the right is of DFSMDA, RECON, or both.

B: The information about DFSMDA and RECON are the same.
M: The information is about DFSMDA.
R: The information is about RECON.

If the information about DFSMDA and RECON are not the same, or if only one of them exists, the column shows M or R.

DFSMDA/RECON
These columns show that the following fields were found both in the DFSMDA dynalloc allocation member and in the DBDS record in the RECON.

DD1/DD2 The ddname of the data set.

DISP The disposition the data set (OLD or SHR). This field is shown only for the DFSMDA dynalloc allocation member.

DSNAME The name of the data set.

FLG Shows whether the following flags in RECON are ON.
I: The image copy needed flag
R: The recovery needed flag
P: The prohibit authorization flag (FP only)

If a flag is OFF, period (.) is shown. If the flag does not exist in the DBDS record, hyphen (-) is shown.

Notes:
1. For HALDBs, this table shows information only for A-side (A-J, L, X) data set groups in the first defined partition.
2. For DEDB databases, this table shows information only for the first defined area data set (ADS) of each area.
PSB Check report
The PSB Check report is generated in the SYSPRINT data set as a result of the consistency check for PSBs.

This report contains the following three parts:
- Library information
- Program Specification Block (PSB) information
- Program Communication Block (PCB) information

The library information part is printed once on the first page. If FAILONLY=YES statement is specified in SYSIN DD, the Program Specification Block information part and the Program Communication Block information part are printed only for the PSBs and the PCBs for which any inconsistency was detected in the consistency check. Otherwise, these parts are printed for all PSBs.

Subsections:
- "Sample report"
- “Report field descriptions for the library information part” on page 149
- “Report field descriptions for the PSB information part” on page 150
- “Report field descriptions for the PCB information part” on page 153

Sample report
The following figure shows an example of the PSB Check report.

---

**IMS LIBRARY INTEGRITY UTILITIES - CONSISTENCY CHECKER**
**5655-U08**
**DATE: 10/01/2015 TIME: 12.04.07**
**FABLECHK - V2.R2**

**LIBRARY INFORMATION**

<table>
<thead>
<tr>
<th>LIBRARY</th>
<th>VERSION</th>
<th>DSNAME</th>
<th>DSNAME</th>
<th>VOLUME</th>
<th>SUFFIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS</td>
<td>13.1</td>
<td></td>
<td></td>
<td>IMSVS</td>
<td></td>
</tr>
<tr>
<td>DFSRESLB</td>
<td>VOLUME=IMSVS DSNAME=IMSVS.DFSRESL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSBLIB</td>
<td>VOLUME=IMSVS DSNAME=IMSVS.PSBLIB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACBLIB</td>
<td>VOLUME=IMSVS DSNAME=IMSVS.ACBLIB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MODLKS</td>
<td>VOLUME=IMSVS DSNAME=IMSVS.MODLKS SUFFIX = A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

*Figure 32. PSB Check report (Part 1 of 2)*
Report field descriptions for the library information part

This part contains information about the input libraries for the PSB check.

**IMS**
IMS version and release number retrieved from the library that is specified in DFSRESLB DD.

The following lines show the volume name and the data set name of each library.

**DFSRESLB**
IMS load library specified in DFSRESLB DD.

**PSBLIB**
PSB library specified in PSBLIB DD.

**ACBLIB**
ACB library specified in ACBLIB DD.
Unless each data set is specified in the DD statement, the line for the library is not printed.

**MODBLKS**
MODBLKS module library specified in MODBLKS DD.
MODBLKS information is not printed in the following cases:
- When DRD=YES is specified in SYSIN DD.
- When the data set is not specified on the DD statement.
- When an IMS catalog PSB is processed.

**SUFFIX = (x)**
Suffix of the MODBLKS module name DFSPDIRx.

**SYSRDDS**
System RDDS specified in SYSRDDS DD
SYSRDDS information is not printed in the following cases:
- When DRD=NO is specified in SYSIN DD.
- When the data set is not specified on the DD statement.
- When an IMS catalog PSB is processed.

**NSYSRDDS**
Non-system RDDS specified in NSYSRDDS DD
NSYSRDDS information is not printed in the following cases:
- When DRD=NO is specified in SYSIN DD.
- When the data set is not specified on the DD statement.
- When an IMS catalog PSB is processed.

**Report field descriptions for the PSB information part**

This part contains information about each application program definition and the results of the consistency check.

In this part, information about only the libraries to be verified is printed.

**PSB NAME**
PSB name. Volume serial number and data set name of the library that contains the PSB member.

The columns of the table in the PSB information part are as follows:

**LIBRARY**
Library that contains each definition.

**CHK**
If any inconsistency is detected between the PSB and each definition, *** is shown.

**ITEM/FIELD**
Definition item or field.

**CONTENTS**
Contents of each field.

**PSBLIB**
Contents of the PSBs field.

The rows of the table in the PSB information part are as follows:

**ACBLIB**
ACB MBR
Whether an ACB member that corresponds to the PSB is found in
the ACB library.

GENDATE
The date and time when the ACB was generated.

Notes:
1. If the generation date of the PSB is later than the generation
date of the ACB, Consistency Checker regards them as inconsistent.
2. If the PSB was generated by IMS Version 3 or higher, the PSB’s
field is shown.

IMSREL
The version and release of the IMS system that generated the ACB.

Notes:
1. If they are different from the version and release of the IMS
system that is defined in the DFSRESLB library, Consistency
Checker regards them as inconsistent.
2. If the PSB was generated by IMS Version 3 or higher, the PSB’s
field is shown.

LANG
The compiler language.

Even if you specify LANG=blank on the PSBGEN statement,
LANG=ASSEM/COBOL is shown.

TPPCB NO
The number of TP PCBs in the PSB.

DBPCB NO
The number of DB PCBs in the PSB.

GSAMPCB NO
The number of GSAM PCBs in the PSB.

MAXQ
The maximum number of database calls with Qx command codes
that can be issued between synchronization points.

SSASIZE
The maximum total length of all SSAs used by the application
program.

Unless SSASIZE is specified on the PSBGEN, Consistency Checker
does not verify this field.

IOEROPN
The condition code that is returned to the operating system when
the IMS system terminates normally, and errors that occurred on
any database while running the application program.

LOCKMAX
The maximum number of locks that an application program can
get at one time.

CMPAT
Whether the PSB is treated as if there were an I/O PCB.
OLIC  Whether the user of the PSB is authorized to run the Online Database Image Copy utility or the Surveyor utility feature that runs as a BMP against a database named in the PSB.

GSROLBOK  Whether an internal ROLB call should be done to roll back non-GSAM database updates.

MODBLKS

PSB DEF  Whether an application program definition that corresponds to the PSB is found in the specified MODBLKS module.

PGM TYPE

The type of application program.

TP:  This value specifies that the IMS system schedules the program when messages processed by the program exist in the system.

BATCH:  This value specifies that the program can use DL/I in the system region of the IMS control program and can refer to the message queues.

MODBLKS information is not printed in the following cases:

• When DRD=YES is specified in SYSIN DD.
• When the data set is not specified on the DD statement.
• When an IMS catalog PSB is processed.

SYSRDDS

PSB DEF  Whether an application program definition that corresponds to the PSB is found in the specified system RDDS.

PGM TYPE

The type of application program.

TP:  This value specifies that the IMS system schedules the program when messages processed by the program exist in the system.

BATCH:  This value specifies that the program can use DL/I in the system region of the IMS control program and can refer to the message queues.

SYSRDDS information is not printed in the following cases:

• When DRD=NO is specified in SYSIN DD.
• When the data set is not specified on the DD statement.
• When an IMS catalog PSB is processed.

NSYSRDDS

PSB DEF  Whether an application program definition that corresponds to the PSB is found in the specified non-system RDDS.

PGM TYPE

The type of application program.
TP: This value specifies that the IMS system schedules the program when messages processed by the program exist in the system.

BATCH: This value specifies that the program can use DL/I in the system region of the IMS control program and can refer to the message queues.

NSYSRDDS information is not printed in the following cases:
• When DRD=NO is specified in SYSIN DD.
• When the data set is not specified on the DD statement.
• When an IMS catalog PSB is processed.

Report field descriptions for the PCB information part

This part contains information about program communication block definitions, and results of the consistency check.

Notes:
1. If you define TP PCBs or DB PCBs in a PSB, the TP PCB information part or the DB PCB information part is shown.
2. When you specify FAILONLY=YES and PCBERRLMT=nnnn, only the inconsistent PCBs up to nnnn in each PSB will be printed, and no consistent PCBs will be shown in this part.

TPPCB INFORMATION

The columns of the table in the TP PCB information part are as follows:

PCB # The sequential number of the TP PCBs.

CHK If any inconsistency is detected between each definition and the PSB, *** is printed. Additionally * is shown on the right side of each data that is determined as inconsistent.

PA Identifies whether the information on the right is of PSBLIB, ACBLIB, or both.

B: The information about PSBLIB and ACBLIB are the same.
P: The information is about PSBLIB.
A: The information is about ACBLIB.

If the information about PSBLIB and ACBLIB are not the same, or if only one of them exists, the column shows P or A.

PSBLIB/ACLIB
These columns show the following fields that are found in the PSB and the ACB:

LTERM/NAME The output message destination.

PCBNAME (label) The name of the PCB or the label for the PCB.

ALTR: ALTRESP Whether the PCB can be used instead of the I/O PCB.

SAME: SAMETRM Whether the IMS system verifies that the logical terminal
named in the response alternate PCB is assigned to the same physical terminal as the logical terminal that originated the input message.

**MODI: MODIFY**
Whether the dynamic modification of the destination name is allowed.

**EXPR: EXPRESS**
Whether messages from the PCB are to be sent or are to be backed out when the application program abends.

**LIST**
Whether the named PCB is included in the PCB list that is passed to the application program at the entry.

**DBPCB INFORMATION**
The columns of the table in the DB PCB information part are as follows:

<table>
<thead>
<tr>
<th>PCB #</th>
<th>The sequential number of the DB PCBs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHK</td>
<td>If any inconsistency is detected between each definition and the PSB, *** is printed. Additionally * is shown on the right side of each data that is determined as inconsistent.</td>
</tr>
<tr>
<td>PA</td>
<td>Identifies whether the information on the right is of PSBLIB, ACBLIB, or both.</td>
</tr>
<tr>
<td></td>
<td>B: The information about PSBLIB and ACBLIB are the same.</td>
</tr>
<tr>
<td></td>
<td>P: The information is about PSBLIB.</td>
</tr>
<tr>
<td></td>
<td>A: The information is about ACBLIB.</td>
</tr>
</tbody>
</table>

If the information about PSBLIB and ACBLIB are not the same, or if only one of them exists, the column shows P or A.

**PSBLIB/ACBLIB**
These columns show that the following fields are found in the PSB and the ACB:

<table>
<thead>
<tr>
<th>DBDNAME</th>
<th>DBD name.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCBNAME (label)</td>
<td>The name of the PCB or the label for the PCB.</td>
</tr>
<tr>
<td>SENSEGS</td>
<td>The number of sensitive segments defined in the PCB.</td>
</tr>
<tr>
<td>KEYLN</td>
<td>The longest concatenated key length.</td>
</tr>
</tbody>
</table>

**Note:** If the key length of the PCB for a DEDB database that is defined in the ACB is the same as that in the PSB adjusted to the fullword boundary, Consistency Checker regards them as consistent.

**PROCOPT**
The processing options on sensitive segments that are specified on the PCB statement.

**Note:** Even if PROCOPT=L is specified in a PCB for an HIDAM or a PHIDAM database, Consistency Checker regards it as PROCOPT=LS.
PROCSEQ(D)
The name of a secondary index that is used to process a database through a secondary processing sequence.

VIEW   Whether applications use MSDB commit view.

LIST   Whether the named PCB is included in the PCB list that is passed to the application program at the entry.
Chapter 5. Multiple Resource Checker utility

The Multiple Resource Checker utility checks the consistency across multiple resources.

Topics:

- “Multiple Resource Checker utility overview” on page 158
- “Checking consistencies with the Multiple Resource Checker utility” on page 160
- “JCL requirements for the Multiple Resource Checker utility” on page 172
- “Control statements for the Multiple Resource Checker utility” on page 175
- “Fields compared in RECON data sets” on page 178
- “JCL examples for the Multiple Resource Checker” on page 182
- “Output from the Multiple Resource Checker utility” on page 190
Multiple Resource Checker utility overview

The Multiple Resource Checker utility checks the consistency across multiple resources, such as across DBDLIBs, PSBLIBs, ACBLIBs, and sets of RECON data sets.

Subtopics:
- "Function overview"
- "Program structure" on page 159
- "Restriction" on page 159
- "Data flow" on page 159

Function overview

The Multiple Resource Checker utility checks the consistency across multiple resources.

If you have multiple DBD libraries, PSB libraries, ACB libraries, and sets of RECON data sets, and you want to identify the libraries and RECON data sets that contain different definitions, run the Multiple Resource Checker utility to generate a Resource Check Summary report. This report is generated in the FABWSUMM data set and contains a matrix table that summarizes any differences in the libraries and RECON data sets.

The Multiple Resource Checker utility can process up to 10 DBDLIBs, PSBLIBs, ACBLIBs, and 10 sets of RECON data sets in one job.

This utility checks each DBD, PSB, and ACB member across libraries and database definitions across multiple sets of RECON data sets. The utility reports that the members or definitions are different when one or more of the following occurrences are detected:

- One or more DBD or PSB members do not exist in one or more libraries
- DBD or PSB members exist in every library, but their definitions are different
- One or more DBD members are not registered in one or more sets of RECON data sets
- DBD members are registered in each set of RECON data sets, but their definitions, such as the database organization type or the DD name, are different from the definitions in the library
- Database definitions and database recovery definitions in the DB record fields or the DBDS record fields are different across sets of RECON data sets

The Multiple Resource Checker utility can also check the consistency across multiple sets of RECON data sets and generate a RECON difference report in the FABWRRPT data set. It checks database definitions and database recovery definitions in RECON record fields, DB record fields, DBDS record fields, and data group record fields.

The utility does not check all of the fields in RECON data sets. For RECON data sets, the utility checks the definitions that relate to the database and, optionally, the definitions that relate to the database recovery environment. Then, the utility reports the results in the Resource Check Summary report and the RECON Difference report. For more information about the fields that are checked, see "Fields compared in RECON data sets" on page 178.
Program structure

The Multiple Resource Checker utility is provided as an MVS batch utility program. Based on the control statements, the program generates a Resource Check Summary report and, optionally, a RECON Difference report, for the specified libraries. This function is invoked by the FABWMCHK load module.

Restriction

The Multiple Resource Checker utility cannot compare database definitions in RECON data sets with ACB members in ACBLIB.

Data flow

The following figure shows the general data flow for Multiple Resource Checker (FABWMCHK).

The input consists of the FABWCTL data set (contains the control statements), RECON data sets, DBDLIB data sets, PSBLIB data sets, and ACBLIB data sets. The output consists of reports and an activity log.

![Data flow for Multiple Resource Checker](image)

*Figure 34. Data flow for Multiple Resource Checker*
Checking consistencies with the Multiple Resource Checker utility

The Multiple Resource Checker utility supports several scenarios for checking the consistency of DBDLIBs, PSBLIBs, ACBLIBs, and sets of RECON data sets.

Checking the consistency of multiple resources

Use the Multiple Resource Checker utility to detect inconsistencies in DBDs or PSBs that are stored in multiple libraries or that are defined in multiple sets of RECON data sets.

About this task

If you have multiple IMS subsystems and each subsystem uses different RECON data sets, DBDLIBs, PSBLIBs, and ACBLIBs, and you want to identify the resources that contain different definitions, run the Multiple Resource Checker to generate a Resource Check Summary report.

The following figure illustrates the resources that are compared in this scenario.

The Multiple Resource Checker utility can process resources that were generated by different IMS releases. To process multiple sets of RECON data sets that are of different IMS releases, the IMS RESLIB of each IMS release is required.

The elapsed time that is required for a job increases as the number of resources to check increases. If you want to check a specific member, you can specify the member name to generate a report for that member. By limiting the number of resources, you can reduce the elapsed time for the job.

Sample JCL for the Multiple Resource Checker utility is in the SHPSJCL0 library, member FABWIVP. You can modify this sample JCL and then use it to run the utility.

Procedure

1. Code the EXEC statement and DD statements for the Multiple Resource Checker utility.
   The following JCL example is for comparing multiple resources:
To check the RECON data sets generated by different IMS releases, code an IMSVnn DD statement for each IMS release.

To check two sets of RECON data sets, code three RECONxxn DD statements for each set of the RECON data sets.

To check DBDs, PSBs, and ACBs in different libraries, specify a DBDLIBxx, PSBLIBxx, ACBLIBxx DD statement for each library.

To generate messages and the Resource Check Summary report, specify the FABWOUT and FABWSUMM DD statements.

To access the RECON data sets, code the SYSPRINT DD statement.

For a description of the JCL statements, see “JCL requirements for the Multiple Resource Checker utility” on page 172.

2. In the FABWCTL data set, code the control statements for the Multiple Resource Checker utility.

To check the resources that are generated by different IMS releases, specify the NOCOMP=IMSREL control statement as follows:

```jcl
//FABWCTL DD *
   NOCOMP=IMSREL
/*
```

For a description of the control statements, see “Control statements for the Multiple Resource Checker utility” on page 175.

3. Submit the job.

4. Examine the Resource Check Summary report that is generated in the FABWSUMM data set.

The following figure shows an example of the Resource Check Summary report.
Figure 36. Report example: Resource Check Summary report for multiple resources

From this report, you can identify that:

- **1** The DBD DBD@0001 exists in all of the specified libraries and in each set of the RECON data sets. The definitions are the same.
- **2** The DBD DBD@0002 exists in all of the specified libraries and in each set of the RECON data sets. However, some of them have different definitions.
- **3** The DBD DBD@0003 is not registered in the RECON data sets and does not exist in the ACB libraries.
- **4** The PSB PSB@0001 exists in all of the specified libraries and its definitions are the same.
- **5** The PSB PSB@0002 exists in all of the specified libraries, but some of them have different definitions.
- **6** The PSB-type ACB PSB@0003 does not exist in one of the ACB libraries.

For a description of the report, see “FABWSUMM data set” on page 190.

**What to do next**

When inconsistent resources are found, you can use IMS Library Integrity Utilities to investigate the differences:

- To check the differences between the members in two libraries, use the DBD/PSB/ACB Compare utility. See Chapter 6, “DBD/PSB/ACB Compare utility,” on page 199.
- To identify the specific RECON field that is different in the RECON data sets, generate a RECON Difference report by using the Multiple Resource Checker utility. See “Checking the consistency of multiple sets of RECON data sets” on page 163.
- To view the database structure, use the DBD/PSB/ACB Mapper utility. See Chapter 7, “DBD/PSB/ACB Mapper utility,” on page 243.

If you have IMS Database Solution Pack or IMS Fast Path Solution Pack, you can also use the DBD/PSB Map Viewer, the IMS Library Integrity Utilities extension for Management Console, to view the database structure and program specifications through a web browser. For more information, see the Management Console User’s Guide.

---

Page 162: IMS Library Integrity Utilities User’s Guide
Checking the consistency of multiple sets of RECON data sets

Use the Multiple Resource Checker utility to check the consistency of definitions across multiple sets of RECON data sets. When inconsistencies are found, you can request an additional report that shows which definitions in the RECON data sets are different.

About this task

If you have multiple sets of RECON data sets and you want to ensure that the definitions in the RECON data sets are consistent, run the Multiple Resource Checker utility to generate a Resource Check Summary report. The Multiple Resource Checker utility checks for inconsistencies in DB record fields and DBDS record fields. If inconsistencies are found, the utility summarizes the differences in the Resource Check Summary report. The Multiple Resource Checker utility also reports the inconsistencies in the RECON record fields, DB record fields, DBDS record fields, and data group record fields in the RECON Difference report.

Restriction: The utility does not check all of the fields in RECON data sets. For the record fields that are checked by the Multiple Resource Checker utility, see “Fields compared in RECON data sets” on page 178.

The following figure illustrates the resources that are compared in this scenario.

Sample JCL for the Multiple Resource Checker utility is in the SHPSJCL0 library, member FABWIVP. You can modify this sample JCL and then use it to run the utility.

Procedure

1. Code the EXEC statement and DD statements for the Multiple Resource Checker utility.
The following JCL example is for comparing multiple sets of RECON data sets:

```
//MLTISTP EXEC PGM=FABWMCHK
//STEPLIB DD DISP=SHR,DSN=HPS.SHPSLMD0
//IMSV11 DD DISP=SHR,DSN=IMS11A.SDFSRESL
//IMSV12 DD DISP=SHR,DSN=IMS12A.SDFSRESL
//IMSV13 DD DISP=SHR,DSN=IMS13A.SDFSRESL
//RECON011 DD DISP=SHR,DSN=IMSVS.TEST.IMS11.RECON011
//RECON012 DD DISP=SHR,DSN=IMSVS.TEST.IMS11.RECON012
//RECON013 DD DISP=SHR,DSN=IMSVS.TEST.IMS11.RECON013
//RECON021 DD DISP=SHR,DSN=IMSVS.TEST.IMS12.RECON021
//RECON022 DD DISP=SHR,DSN=IMSVS.TEST.IMS12.RECON022
//RECON023 DD DISP=SHR,DSN=IMSVS.TEST.IMS12.RECON023
//RECON031 DD DISP=SHR,DSN=IMSVS.TEST.IMS13.RECON031
//RECON032 DD DISP=SHR,DSN=IMSVS.TEST.IMS13.RECON032
//RECON033 DD DISP=SHR,DSN=IMSVS.TEST.IMS13.RECON033
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//FABWRPT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
```

- **1** To check the RECON data sets generated by different IMS releases, code an IMSVnm DD statement for each IMS release.
- **2** To check three sets of RECON data sets, code three RECONxnn DD statements for each set of the RECON data sets.
- **3** To generate messages and the Resource Check Summary report, specify the FABWOUT and FABWSUMM DD statements.
- **4** To generate the RECON Difference report, specify the FABWRPT DD statement.
- **5** To access the RECON data sets, code the SYSPRINT DD statement.

For a description of the JCL statements, see "JCL requirements for the Multiple Resource Checker utility" on page 172.

2. In the FABWCTL data set, code the control statements for the Multiple Resource Checker utility.

To generate the RECON Difference report, specify DIFFREP=YES control statement as follows:

```
//FABWCTL DD *
NOCOMP=IMSREL
DIFFREP=YES
/*
```

For a description of the control statements, see "Control statements for the Multiple Resource Checker utility" on page 175.

3. Submit the job.

4. Examine the Resource Check Summary report that is generated in the FABWSUMM data set. If a RECON Difference report is generated in the FABWRPT data set, also examine that report.

The following figure shows an example of the Resource Check Summary report.
From this report, you can identify whether the database definitions in the DB record fields and DBDS record fields are the same across the sets of RECON data sets.

- **1** The database definition for DBD00001 is registered in all the specified RECON data sets. However, the definitions are different in some RECON data sets.

- **2** The database definition for DBD00002 is registered in all the specified RECON data sets. The definitions in all the RECON data sets are the same.

The following figure shows an example of the RECON Difference report.

![Figure 38](image1)

From this report, you can identify whether the database definitions in the DB record fields and DBDS record fields are the same across the sets of RECON data sets.

- **1** The database definition for DBD00001 is registered in all the specified RECON data sets. However, the definitions are different in some RECON data sets.

- **2** The database definition for DBD00002 is registered in all the specified RECON data sets. The definitions in all the RECON data sets are the same.

The following figure shows an example of the RECON Difference report.

![Figure 39](image2)

From this report, you can identify that in the DB record fields, the value of SHARE LEVEL differs for DBD DBD00001 **1**.

For a description of the reports, see the following topics:

- **“FABWSUMM data set” on page 190**
- **“FABWRRPT data set” on page 194**
Checking the consistency across two resource types

Use the Multiple Resource Checker utility to check the consistency of DBDs between DBD libraries and ACB libraries, or PSBs between PSB libraries and ACB libraries.

About this task

You can use the Multiple Resource Checker utility to check the consistency of DBD or PSB members between multiple DBD or PSB libraries and multiple ACB libraries in one job.

If you want to check the consistency of a specific DBD or PSB member in multiple libraries, you can specify the member name on the DBD= or the PSB= control statement. Doing so improves the performance of the job.

The following figure illustrates the resources that are compared in this scenario.

![Diagram illustrating resources compared in this scenario]

Figure 40. Checking the consistency of DBDs in DBD libraries and ACB libraries

Sample JCL for the Multiple Resource Checker utility is in the SHPSJCL0 library, member FABWIVP. You can modify this sample JCL and then use it to run the utility.

Procedure

1. Code the EXEC statement and DD statements for the Multiple Resource Checker utility.
   The following JCL example is for checking the consistency of DBDs and ACBs in multiple libraries:
To check the DBD members in DBD libraries, specify a DBDLIBxx DD statement for each DBD library.

To check the DBD-type ACB members in ACB libraries, specify an ACBLIBxx DD statement for each ACB library.

To generate messages and the Resource Check Summary report, specify the FABWOUT and FABWSUMM DD statements.

For a description of the JCL statements, see “JCL requirements for the Multiple Resource Checker utility” on page 172.

2. In the FABWCTL data set, code the control statements for the Multiple Resource Checker utility.

If you do not specify control statements, all the members in the specified libraries are checked.

For a description of the control statements, see “Control statements for the Multiple Resource Checker utility” on page 175.

3. Submit the job.

4. Examine the Resource Check Summary report that is generated in the FABWSUMM data set.

The following figure shows an example of the Resource Check Summary report.

Figure 41. Report example: Resource Check Summary report for DBDs in DBD and ACB libraries

From this report, you can identify that:

- 1 The DBD DBD00001 exists in all of the specified libraries and its definitions are the same.

- 2 The DBD DBD00002 exists in all of the specified libraries, but some of them have different definitions.

- 3 The DBD-type ACB DBD00003 does not exist in two ACB libraries.
For a description of the report, see “FABWSUMM data set” on page 190.

What to do next

When inconsistent DBD or PSB members are found, you can use IMS Library Integrity Utilities to investigate the differences:

- To check the differences between the members in two libraries, use the DBD/PSB/ACB Compare utility. See Chapter 6, “DBD/PSB/ACB Compare utility,” on page 199.

- To view the database structure, use the DBD/PSB/ACB Mapper utility. See Chapter 7, “DBD/PSB/ACB Mapper utility,” on page 243.

If you have IMS Database Solution Pack or IMS Fast Path Solution Pack, you can also use the DBD/PSB Map Viewer, the IMS Library Integrity Utilities extension for Management Console, to view the database structure and program specifications through a web browser. For more information, see the Management Console User’s Guide.
Checking the consistency of same resource-type members in multiple libraries

Use the Multiple Resource Checker utility to check the consistency of a DBD or a PSB across multiple libraries.

About this task

You can use the Multiple Resource Checker utility to check the consistency of the same resource-type members across multiple libraries in one job.

The following figure illustrates the resources that are compared in this scenario.

Sample JCL for the Multiple Resource Checker utility is in the SHPSJCL0 library, member FABWIVP. You can modify this sample JCL and then use it to run the utility.

Procedure

1. Code the EXEC statement, DD statements, and control statements for the Multiple Resource Checker utility.

   The following JCL example is for comparing DBD members in multiple DBD libraries:
To check the DBD members in DBD libraries, specify a DBDLIBxx DD statement for each DBD library.

To generate messages and the Resource Check Summary report, specify the FABWOUT and FABWSUMM DD statements.

To check only the DBD libraries, specify CHKONLY=DBD. In this example, ACB libraries are also specified by the ACBLIBxx DD statements. The CHKONLY=DBD specification causes the utility to ignore the ACB libraries that are specified by the ACBLIBxx DD statements. By using this keyword, you can check only the libraries you want without changing the DD statements.

See the following topics to code the JCL statements and control statements:

- "JCL requirements for the Multiple Resource Checker utility" on page 172
- "Control statements for the Multiple Resource Checker utility" on page 175

2. Submit the job.
3. Examine the Resource Check Summary report that is generated in the FABWSUMM data set.

The following figure shows an example of the Resource Check Summary report.

Figure 43. Report example: Resource Check Summary report for a single resource
From this report, you can identify that:

1. The DBDs DBD@0001, DBD@0007, and DBD@0008 exist in all of the specified libraries and their definitions are the same.

2. The DBDs DBD@0005 and DBD@0006 exist in all of the specified libraries, but some of them have different definitions.

3. The DBDs DBD@0002, DBD@0003, and DBD@0004 do not exist in some DBD libraries.

For a description of the report, see “FABWSUMM data set” on page 190.

**What to do next**

When inconsistent DBD or PSB members are found, you can use IMS Library Integrity Utilities to investigate the differences:

- To check the differences between the members in two libraries, use the DBD/PSB/ACB Compare utility. See Chapter 6, “DBD/PSB/ACB Compare utility,” on page 199.

- To view the database structure, use the DBD/PSB/ACB Mapper utility. See Chapter 7, “DBD/PSB/ACB Mapper utility,” on page 243.

If you have IMS Database Solution Pack or IMS Fast Path Solution Pack, you can also use the DBD/PSB Map Viewer, the IMS Library Integrity Utilities extension for Management Console, to view the database structure and program specifications through a web browser. For more information, see the Management Console User’s Guide.
JCL requirements for the Multiple Resource Checker utility

When you code a JCL job to run the Multiple Resource Checker utility, include the EXEC statement and appropriate DD statements.

Subsections:
- "JCL example"
- "EXEC statement"
- "DD statements" on page 173

JCL example

The following figure shows a JCL example that contains the JCL statements for checking the consistency of multiple resources and reporting the differences in multiple sets of RECON data sets.

```plaintext
//stepname EXEC PGM=FABWMCHK
//STEPLIB DD DSN=HPS.SHPSSLMD0,DISP=SHR
//IMSV11 DD DSN=IMSV11.RESLIB,DISP=SHR
//IMSV12 DD DSN=IMSV12.RESLIB,DISP=SHR
//IMSV13 DD DSN=IMSV13.RESLIB,DISP=SHR
//RECON011 DD DSN=IMSVS1.RECON1,DISP=SHR
//RECON012 DD DSN=IMSVS1.RECON2,DISP=SHR
//RECON013 DD DSN=IMSVS1.RECON3,DISP=SHR
//RECON021 DD DSN=IMSVS2.RECON1,DISP=SHR
//RECON022 DD DSN=IMSVS2.RECON2,DISP=SHR
//RECON023 DD DSN=IMSVS2.RECON3,DISP=SHR
//RECON031 DD DSN=IMSVS3.RECON1,DISP=SHR
//RECON032 DD DSN=IMSVS3.RECON2,DISP=SHR
//RECON033 DD DSN=IMSVS3.RECON3,DISP=SHR
//DBDLIB01 DD DSN=IMSVS.DBDLIB1,DISP=SHR
//DBDLIB02 DD DSN=IMSVS.DBDLIB2,DISP=SHR
//DBDLIB03 DD DSN=IMSVS.DBDLIB3,DISP=SHR
//DBDLIB04 DD DSN=IMSVS.DBDLIB4,DISP=SHR
//PSBLIB01 DD DSN=IMSVS.PSBLIB1,DISP=SHR
//PSBLIB02 DD DSN=IMSVS.PSBLIB2,DISP=SHR
//ACBLIB01 DD DSN=IMSVS.ACBLIB1,DISP=SHR
//ACBLIB02 DD DSN=IMSVS.ACBLIB2,DISP=SHR
//ACBLIB03 DD DSN=IMSVS.ACBLIB3,DISP=SHR
//ACBLIB04 DD DSN=IMSVS.ACBLIB4,DISP=SHR
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//FABWRRPT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//FABWCTL DD *
   /* (control statements)
   */
```

Figure 44. JCL for checking the consistency of multiple resources and the difference in multiple sets of RECON data sets

EXEC statement

The EXEC statement must be in the following format:

```plaintext
//stepname EXEC PGM=FABWMCHK
```

You can optionally specify the IMSPLEX and DBRCGRP parameters on the EXEC statement. If specified, the utility processes RECON data sets that belong to the
same group identified by the IMSLEX and DBRCGRP parameters. The utility cannot process the RECON data sets that do not belong to the group.

For example:

```
//stepname EXEC PGM=FABWMCHK,PARM='IMSPLEX=imsplex, DBRCGRP=dbrgrp'
```

**IMSPLEX=imsplex**  
A 1-to-5 character IMSplex name to be used for RECON data sets.

**DBRCGRP=dbrgrp**  
A 1-to-3 character identifier (ID) assigned to a group of DBRC instances that access the same RECON data set in an IMSplex.

### DD statements

Code the following DD statements to identify the source of input and the placement of output information:

**STEPLIB DD or JOBLIB DD**  
Required. This input DD statement defines the IMS Library Integrity Utilities load module library.

**IMSVnn DD**  
Optional. This input DD statement points to the library that contains the IMS load modules. Specify this DD statement if you want to compare RECON data sets.

For *nn*, specify the version of IMS. For example, to process the RECON data sets that were generated by IMS Version 12, specify an IMSV12 DD statement.

If you use the DBRC Structured Call Interface (SCI) Registration Exit to access RECON data sets, specify it in this DD statement. If specified, the utility processes the RECON data sets that can be accessed with the SCI exit.

**RECONxxn DD**  
Optional. This input DD statement points to the RECON data sets.

*xx* Specify 01 to process one set of RECON data sets. To process multiple sets of RECON data sets, specify a sequential number for each set of the RECON data sets. Up to 10 sets can be specified.

*n* Specify 1, 2, or 3 to identify the RECON data set within a set of RECON data sets:

1. Copy1 RECON data set.
2. Copy2 RECON data set.

**DBDLIBxx DD**  
Optional. This input data set points to the library that contains the DBDs to check. For *xx*, assign a sequential number for up to 10 libraries.

Concatenation of multiple DBD libraries is not supported.

**PSBLIBxx DD**  
Optional. This input data set points to the library that contains the PSBs to check. For *xx*, assign a sequential number for up to 10 libraries.

Concatenation of multiple PSB libraries is not supported.
ACBLIBxx DD  
Optional. This input data set points to the library that contains the ACBs to check. For xx, assign a sequential number for up to 10 libraries.

Concatenation of multiple ACB libraries is not supported.

FABWOUT DD  
Required. All the input parameters, runtime parameters, and error messages are written to this output data set.

The record format is fixed block (FB). The logical record length is 133.
Block size, if coded, must be a multiple of 133.

FABWSUMM DD  
Required. The Resource Check Summary report is generated in this output data set.

The record format is fixed block (FB). The logical record length is 133.
Block size, if coded, must be a multiple of 133.

FABWRRPT DD  
Optional. The RECON Difference report is generated in this output data set. If you specify DIFFREP=Y on the FABWCTL statement, which requests to generate the RECON Difference report, you must specify this DD statement.

The record format is fixed block (FB). The logical record length is 133.
Block size, if coded, must be a multiple of 133.

FABWCTL DD  
Optional. This input data set contains the control statements for the FABWMCHK program.

The record format is fixed block (FB). The logical record length is 80. Block size, if coded, must be a multiple of 80.

For a complete description of the control statements, see "Control statements for the Multiple Resource Checker utility" on page 175.

SYSPRINT DD  
Optional. This data set is used when the utility accesses RECON data sets. If you specify the RECONxxn DD statements, specify this DD statement.

The record format is fixed block (FB). The logical record length is 133.
Block size, if coded, must be a multiple of 133.

The following messages are written to this data set until the utility finds a valid combination of IMS versions in IMS DD and RECON DD statements:

DSP0024I  RECON(n) HEADER RECORD MISSING OR INVALID
DSP0245I  JOB TERMINATED DUE TO UNAVAILABLE RECON DATA SETS
Control statements for the Multiple Resource Checker utility

The input to the Multiple Resource Checker utility consists of control statements in the FABWCTL data set. These control statements contain keywords that specify the function and the names of the DBDs, PSBs, or ACBs to check.

If the FABWCTL data set is not specified, the default options are used in the job, and all the members and databases in the specified libraries and RECON data sets are checked.

You can specify up to 9999 control statements in the FABWCTL statement.

This data set usually resides in the input stream. However, it can be defined as a sequential data set or as a member of a partitioned data set. It must contain one 80-byte fixed-length record for each control statement. Block size, if coded, must be a multiple of 80.

A Resource Check Summary report is always generated. In the report, the members are reported in alphabetical order, regardless of the order they are specified in the control statements.

Subsections:

• “Control statement example”
• “Syntax rules”
• “Control statement keywords” on page 176

Control statement example

Control statements can be coded as follows:

```
//FABWCTL DD *
    NOCOMP=parameter,parameter
    CHKONLY=parameter,parameter
    CHKRECON=parameter
    DIFFREP=parameter
    DBD=member
/*
```

Syntax rules

The control statements for Multiple Resource Checker must adhere to the following syntax rules:

• Control statements can be coded on any columns in the range of 2 - 80.
• In the control statement fields, keywords, equal signs (=), and member names must not be separated by blanks. Because a blank serves as the delimiter, only a comment can be written after a blank.
• A statement with an asterisk (*) in column 1 is treated as a comment.
• The control statements can be specified in any order. For example:

```
DBD=member
CHKONLY=parameter
CHKRECON=parameter
NOCOMP=parameter
DIFFREP=parameter
```
Control statement keywords

The following keywords are supported:

**CHKONLY=** *parameter*
This keyword causes the utility to check only the specific resources.

For example, if you specify DBDLIBxx DD and ACBLIBxx DD statements and CHKONLY=DBD,ACB, the utility checks the consistency of DBD members across the DBD libraries and the consistency of ACB members across the ACB libraries. Without the CHKONLY control statement, the utility checks the consistency between the DBD libraries and the ACB libraries.

**DBD**
Compares the members in the libraries that are specified by the DBDLIBxx DD statements.

**PSB**
Compares the members in the libraries that are specified by the PSBLIBxx DD statements.

**ACB**
Compares the members in the libraries that are specified by the ACBLIBxx DD statements.

**RECON**
Compares the database definitions in the RECON data sets that are specified by the RECONxxn DD statements.

**CHKRECON=DBDEF | DBDEF_RCV**
Specifies the scope of the fields to compare when the utility checks the consistency of definitions across multiple sets of RECON data sets.

**DBDEF**
Compares the fields that relate to database definitions.

**CHKRECON=DBDEF** is the default value.

**DBDEF_RCV**
Compares the fields that relate to database definitions and the fields that relate to the database recovery environment.

For a list of the record fields that are compared by this option, see [“Fields compared in RECON data sets” on page 178](#).

**NOCOMP=** *parameter*
Specifies the fields that you do not want to check.

**IMSREL**
The utility compares the DBD, PSB, ACB members that are generated by the DBDGEN, PSBGEN, or ACBGEN utility of different IMS release levels or sets of RECON data sets that are generated by different IMS release levels. If you specify this parameter, IMS release levels are not checked. Also the fields that were added or deleted in a higher release of IMS are not checked.

The DBD and PSB fields that are not checked when **NOCOMP=** IMSREL is specified are the same as the fields that are not compared by the **DBD/PSB/ACB Compare utility**. For such fields, see [“NOCOMP control statement” on page 213](#).

**VERSION**
The field that is related to the **VERSION=** statement of the DBDGEN utility in DBD and ACB members is not checked.
METADATA

The metadata field in DBDs, PSBs, and ACBs are not checked.

Instead of specifying NOCOMP=METADATA, you can specify NOCOMP=CATALOG. CATALOG is an alias for METADATA.

DIFFREP=YES | NO

Specifies whether to generate the RECON Difference report. This keyword can be specified only once.

- YES Generate the RECON Difference report when inconsistencies are found in the RECON data sets.
- NO Do not generate the RECON Difference report. DIFFREP=NO is the default value.

DBD=member

The name of the member to check.

PSB=member

The name of the member to check.

For DBD= and PSB= keywords, you can specify a wildcard in any position of a character string. The asterisk (*) and the percent sign (%) are supported as wildcard characters. An asterisk represents 0 - 8 characters, and the percent sign represents a single character. If two or more asterisks are specified sequentially, only the first asterisk is recognized.
Fields compared in RECON data sets

When the Multiple Resource Checker utility checks the consistency of database definitions across multiple sets of RECON data sets, the record fields that are compared depend on the parameter that is specified for the CHKRECON keyword in the FABWCTL control statement.

- If you specify CHKRECON=DBDEF, the fields that relate to database definitions are compared.
- If you specify CHKRECON=DBDEF_RCV, the fields that relate to database definitions and database recovery environment definitions are compared. For the fields that relate to the database recovery environment definitions, the utility compares only the fields that IMS does not change.

The following tables show the fields that are compared by the Multiple Resource Checker utility.

Note: To learn more about these fields, see the following topics in IMS Commands:
- "Data group record fields"
- "DB (IMS) record fields"
- "DB (HALDB) record fields"
- "DB (PART) record fields"
- "DB (Fast Path) record fields"
- "DBDS (non-Fast Path) record fields"
- "DBDS (Fast Path) record fields"

Table 5. DB (IMS) record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON= DBDEF</th>
<th>CHKRECON= DBDEF_RCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD=dbdname</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>TYPE=IMS</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>SHARE LEVEL=n</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DBRCVGRP=rcvgrpnm</td>
<td></td>
<td>Compared</td>
</tr>
<tr>
<td>FLAG: RECOVERABLE=YES</td>
<td>NO</td>
<td>Compared</td>
</tr>
<tr>
<td>FLAG: DATABASE LEVEL TRACK=YES</td>
<td>NO</td>
<td>Compared</td>
</tr>
<tr>
<td>IC NEEDED DISABLED</td>
<td></td>
<td>Compared</td>
</tr>
</tbody>
</table>

Table 6. DB (HALDB) record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON= DBDEF</th>
<th>CHKRECON= DBDEF_RCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD=dbdname</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>TYPE=HALDB</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>SHARE LEVEL=n</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DBRCVGRP=rcvgrpnm</td>
<td></td>
<td>Compared</td>
</tr>
<tr>
<td>PSNAME=psname</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DBORG=dbaseorg</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DSORG=dsetorg</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>RECOVERABLE=YES</td>
<td>NO</td>
<td>Compared</td>
</tr>
<tr>
<td>PARTITIONS=n</td>
<td>Compared</td>
<td>Compared</td>
</tr>
</tbody>
</table>
### Table 6. DB (HALDB) record fields (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON= DBDEF</th>
<th>CHKRECON= DBDEF_RCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>ONLINE REORG CAPABLE</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DATA SET GROUP MEMBERS</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>IC NEEDED DISABLED</td>
<td>-</td>
<td>Compared</td>
</tr>
</tbody>
</table>

### Table 7. DB (PART) record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON= DBDEF</th>
<th>CHKRECON= DBDEF_RCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD=dbdname</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>MASTER DB=HALDB master name</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>TYPE=PART</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DSN PREFIX=dsname</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>PARTITION ID=nnnnnn</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>REORG# = nnnnn</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>RANDOMIZER: NAME=ANCHOR=</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>HIGH BLOCK#=BYTES=nnnnnnnn</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>FREE SPACE: FREE BLOCK FREQ FACTOR=nn</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>FREE SPACE PERCENTAGE=nn</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>PARTITION HIGH KEY/STRING (CHAR):  (LENGTH=NNN)</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>PARTITION HIGH KEY/STRING (HEX):</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>OSAM BLOCK SIZE: s = nnnnn</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>ALTER BLOCK SIZE: s = nnnnn</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DATABASE LEVEL TRACK=YES</td>
<td>NO</td>
<td>Compared</td>
</tr>
<tr>
<td>PARTITION DISABLED=YES</td>
<td>NO</td>
<td>Compared</td>
</tr>
<tr>
<td>ONLINE REORG CAPABLE=YES</td>
<td>NO</td>
<td>Compared</td>
</tr>
</tbody>
</table>

### Table 8. DB (FP) record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON= DBDEF</th>
<th>CHKRECON= DBDEF_RCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD=dbdname</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>TYPE=FP</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>SHARE LEVEL=n</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>RECOVERABLE=YES</td>
<td>NO</td>
<td>Compared</td>
</tr>
<tr>
<td>FULLSEG DEFAULT=YES</td>
<td>NO</td>
<td>Compared</td>
</tr>
</tbody>
</table>

### Table 9. DBDS (non-FP) record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON= DBDEF</th>
<th>CHKRECON= DBDEF_RCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSN=dsname</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>TYPE=IMS</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>TYPE=PART</td>
<td>Compared</td>
<td>Compared</td>
</tr>
</tbody>
</table>
### Table 9. DBDS (non-FP) record fields (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON=</th>
<th>CHKRECON=</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD=dbname</td>
<td>DBDEF</td>
<td>DBDEF_RCV</td>
</tr>
<tr>
<td>DDN=ddname</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DBORG=dbaseorg</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DSORG=dsetorg</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>CAGRP=cagrpname</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>GENMAX=nnnn</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>REUSE</td>
<td>NOREUSE</td>
<td>-</td>
</tr>
<tr>
<td>RECOVPD=nnn</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>DEFTJCL=member</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>ICJCL=member</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>OICJCL=member</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>RECOVJCL=member</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>RECJCL=member</td>
<td>-</td>
<td>Compared</td>
</tr>
</tbody>
</table>

### Table 10. DBDS (FP) record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON=</th>
<th>CHKRECON=</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD=dbname</td>
<td>DBDEF</td>
<td>DBDEF_RCV</td>
</tr>
<tr>
<td>AREA=areaname</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>TYPE=FP</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>SHARE LEVEL=n</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>BORG=dbaseorg</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>DSORG=dsetorg</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>CAGRP=cagrpname</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>GENMAX=nnnn</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>REUSE</td>
<td>NOREUSE</td>
<td>-</td>
</tr>
<tr>
<td>RECOVPD=nnn</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>VSO</td>
<td>NOVSO</td>
<td>Compared</td>
</tr>
<tr>
<td>PREOPEN</td>
<td>NOPREOPEN</td>
<td>Compared</td>
</tr>
<tr>
<td>PRELOAD</td>
<td>NOPRELOAD</td>
<td>Compared</td>
</tr>
<tr>
<td>FULLSEG</td>
<td>NOFULLSG</td>
<td>Compared</td>
</tr>
<tr>
<td>CFSTR1=cfstr_name</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>CFSTR2=cfstr_name</td>
<td>Compared</td>
<td>Compared</td>
</tr>
<tr>
<td>LKASID</td>
<td>NOLKASID</td>
<td>Compared</td>
</tr>
<tr>
<td>MAS</td>
<td>NOMAS</td>
<td>Compared</td>
</tr>
<tr>
<td>DEFTJCL=member</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>ICJCL=member</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>RECOVJCL=member</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>DBRCVGRP=rcvgrpnm</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>DATABASE LEVEL TRACK=</td>
<td>YES</td>
<td>NO</td>
</tr>
</tbody>
</table>
The fields in the following two tables are compared only when FABWRRPT DD is specified.

Table 11. RECON record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON=DBDEF</th>
<th>CHKRECON=DBDEF_RCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVERY CONTROL DATA SET, IMS VxRx</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>FORCER</td>
<td>NOFORCER</td>
<td>-</td>
</tr>
<tr>
<td>LOG DSN CHECK=xxxxxxx</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>STARTNEW=YES</td>
<td>NO</td>
<td>-</td>
</tr>
<tr>
<td>TRACEON</td>
<td>TRACEOFF</td>
<td>-</td>
</tr>
<tr>
<td>SSID=xxxxxxx</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>LISTDLOG=YES</td>
<td>NO</td>
<td>-</td>
</tr>
<tr>
<td>CA</td>
<td>IC</td>
<td>LOG DATA SETS</td>
</tr>
<tr>
<td>CATALOGED=YES</td>
<td>NO</td>
<td>-</td>
</tr>
<tr>
<td>LOG RETENTION PERIOD=yy.ddd hh:mm:ss.t</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>COMMAND AUTH=SAF</td>
<td>EXIT</td>
<td>BOTH</td>
</tr>
<tr>
<td>HLQ=hlq name</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>RCNQUAL=data_set_name</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>CATALOG=catalog_name</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>ACCESS=SERIAL</td>
<td>PARALLEL</td>
<td>-</td>
</tr>
<tr>
<td>LIST=STATIC</td>
<td>CONCURR</td>
<td>-</td>
</tr>
<tr>
<td>SIZALERT=xxxxxxxxxxxx xxxxxxxxxxxxx</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>LOGALER=xxxxxxxxxxxx xxxxxxxxxxxxx</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>REORG NUMBER VERIFICATION=YES</td>
<td>NO</td>
<td>-</td>
</tr>
<tr>
<td>IMSPLEX=imsplex_name, GROUP ID=group_ID</td>
<td>-</td>
<td>Compared</td>
</tr>
</tbody>
</table>

Table 12. Data group record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>CHKRECON=DBDEF</th>
<th>CHKRECON=DBDEF_RCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRPNAME=grpname</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>#MEMBERS=nnn</td>
<td>-</td>
<td>Compared</td>
</tr>
<tr>
<td>dbname ddname/arameame</td>
<td>-</td>
<td>Compared</td>
</tr>
</tbody>
</table>
JCL examples for the Multiple Resource Checker

The following topics provide JCL examples for running the Multiple Resource Checker utility.

Examples: Checking the consistency of multiple resources

Use the following examples to check the consistency of multiple resources with the Multiple Resource Checker utility.

Example: Checking the consistency of database definitions across multiple libraries and RECON data sets

The following JCL example is for checking the consistency of DBD members and DBD-type ACB members in DBDLIBs and ACBLIBs. This JCL job also checks the consistency of database definitions across multiple sets of RECON data sets.

The DBD= keywords specify the DBD members to check.

```
//stepname EXEC PGM=FABWCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//IMSV11 DD DSN=IMSV11.RESLIB,DISP=SHR
//IMSV12 DD DSN=IMSV12.RESLIB,DISP=SHR
//IMSV13 DD DSN=IMSV13.RESLIB,DISP=SHR
//RECON011 DD DSN=IMSVS1.RECON1,DISP=SHR
//RECON012 DD DSN=IMSVS1.RECON2,DISP=SHR
//RECON013 DD DSN=IMSVS1.RECON3,DISP=SHR
//RECON021 DD DSN=IMSVS2.RECON1,DISP=SHR
//RECON022 DD DSN=IMSVS2.RECON2,DISP=SHR
//RECON023 DD DSN=IMSVS2.RECON3,DISP=SHR
//RECON031 DD DSN=IMSVS3.RECON1,DISP=SHR
//RECON032 DD DSN=IMSVS3.RECON2,DISP=SHR
//RECON033 DD DSN=IMSVS3.RECON3,DISP=SHR
//DBDLIB01 DD DSN=IMSVS.DBDLIB1,DISP=SHR
//DBDLIB02 DD DSN=IMSVS.DBDLIB2,DISP=SHR
//DBDLIB03 DD DSN=IMSVS.DBDLIB3,DISP=SHR
//DBDLIB04 DD DSN=IMSVS.DBDLIB4,DISP=SHR
//ACBLIB01 DD DSN=IMSVS.ACBLIB1,DISP=SHR
//ACBLIB02 DD DSN=IMSVS.ACBLIB2,DISP=SHR
//ACBLIB03 DD DSN=IMSVS.ACBLIB3,DISP=SHR
//ACBLIB04 DD DSN=IMSVS.ACBLIB4,DISP=SHR
//ACBLIB05 DD DSN=IMSVS.ACBLIB5,DISP=SHR
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//FABWCTL DD *
  DBD=DBD001
  DBD=DBD002
  DBD=DBD003
/*
```

Figure 45. Example for checking the consistency of database definitions across multiple libraries and RECON data sets
Example: Checking the consistency of DBDs and DBD-type ACBs

The following JCL example is for checking the consistency of DBD members in DBDLIBs and DBD-type ACB members in ACBLIBs.

The DBD= keywords specify the DBD or DBD-type ACB members to check.

```
//stepname EXEC PGM=FABWMCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB01 DD DSN=IMSVS.DBDLIB1,DISP=SHR
//DBDLIB02 DD DSN=IMSVS.DBDLIB2,DISP=SHR
//DBDLIB03 DD DSN=IMSVS.DBDLIB3,DISP=SHR
//DBDLIB04 DD DSN=IMSVS.DBDLIB4,DISP=SHR
//DBDLIB05 DD DSN=IMSVS.DBDLIB5,DISP=SHR
//DBDLIB06 DD DSN=IMSVS.DBDLIB6,DISP=SHR
//DBDLIB07 DD DSN=IMSVS.DBDLIB7,DISP=SHR
//DBDLIB08 DD DSN=IMSVS.DBDLIB8,DISP=SHR
//DBDLIB09 DD DSN=IMSVS.DBDLIB9,DISP=SHR
//DBDLIB10 DD DSN=IMSVS.DBDLIBA,DISP=SHR
//ACBLIB01 DD DSN=IMSVS.ACBLIB1,DISP=SHR
//ACBLIB02 DD DSN=IMSVS.ACBLIB2,DISP=SHR
//ACBLIB03 DD DSN=IMSVS.ACBLIB3,DISP=SHR
//ACBLIB04 DD DSN=IMSVS.ACBLIB4,DISP=SHR
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//FABWCTL DD *
  DBD=DBD0001
  DBD=DBD0002
  DBD=DBD0003
  DBD=DBD0004
/*
```

Figure 46. Example for checking the consistency of DBDs and DBD-type ACBs
**Example: Checking the consistency of PSBs and PSB-type ACBs**

The following JCL example is for checking the consistency of PSB members in PSBLIBs and PSB-type ACB members in ACBLIBs.

The PSB= keywords specify the PSB members or PSB-type ACB members to check.

```
//stepname EXEC PGM=FABWCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//PSBLIB01 DD DSN=IMSVS.PSBLIB1,DISP=SHR
//PSBLIB02 DD DSN=IMSVS.PSBLIB2,DISP=SHR
//PSBLIB03 DD DSN=IMSVS.PSBLIB3,DISP=SHR
//PSBLIB04 DD DSN=IMSVS.PSBLIB4,DISP=SHR
//PSBLIB05 DD DSN=IMSVS.PSBLIB5,DISP=SHR
//PSBLIB06 DD DSN=IMSVS.PSBLIB6,DISP=SHR
//PSBLIB07 DD DSN=IMSVS.PSBLIB7,DISP=SHR
//PSBLIB08 DD DSN=IMSVS.PSBLIB8,DISP=SHR
//PSBLIB09 DD DSN=IMSVS.PSBLIB9,DISP=SHR
//PSBLIB10 DD DSN=IMSVS.PSBLIBA,DISP=SHR
//ACBLIB01 DD DSN=IMSVS.ACBLIB1,DISP=SHR
//ACBLIB02 DD DSN=IMSVS.ACBLIB2,DISP=SHR
//ACBLIB03 DD DSN=IMSVS.ACBLIB3,DISP=SHR
//ACBLIB04 DD DSN=IMSVS.ACBLIB4,DISP=SHR
//ACBLIB05 DD DSN=IMSVS.ACBLIB5,DISP=SHR
//ACBLIB06 DD DSN=IMSVS.ACBLIB6,DISP=SHR
//ACBLIB07 DD DSN=IMSVS.ACBLIB7,DISP=SHR
//ACBLIB08 DD DSN=IMSVS.ACBLIB8,DISP=SHR
//ACBLIB09 DD DSN=IMSVS.ACBLIB9,DISP=SHR
//ACBLIB10 DD DSN=IMSVS.ACBLIBA,DISP=SHR
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//FABWCTL DD *
//PSB=PSB0001
//PSB=PSB0002
//PSB=PSB0003
//PSB=PSB0004
//PSB=PSB0005
/*
```

*Figure 47. Example for checking the consistency of PSBs and PSB-type ACBs*
Example: Checking the consistency of database definitions across multiple sets of RECON data sets

The following JCL example is for checking the consistency between 10 sets of RECON data sets.

The utility checks the consistency of database definitions in the DB record fields and the DBDS record fields.

```jcl
//stepname EXEC PGM=FABWMCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//IMSV11 DD DSN=IMSV11.RESLIB,DISP=SHR
//IMSV12 DD DSN=IMSV12.RESLIB,DISP=SHR
//IMSV13 DD DSN=IMSV13.RESLIB,DISP=SHR
//RECON011 DD DSN=IMSVS1.RECON1,DISP=SHR
//RECON012 DD DSN=IMSVS1.RECON2,DISP=SHR
//RECON013 DD DSN=IMSVS1.RECON3,DISP=SHR
//RECON021 DD DSN=IMSVS2.RECON1,DISP=SHR
//RECON022 DD DSN=IMSVS2.RECON2,DISP=SHR
//RECON023 DD DSN=IMSVS2.RECON3,DISP=SHR
//RECON031 DD DSN=IMSVS3.RECON1,DISP=SHR
//RECON032 DD DSN=IMSVS3.RECON2,DISP=SHR
//RECON033 DD DSN=IMSVS3.RECON3,DISP=SHR
//RECON041 DD DSN=IMSVS4.RECON1,DISP=SHR
//RECON042 DD DSN=IMSVS4.RECON2,DISP=SHR
//RECON043 DD DSN=IMSVS4.RECON3,DISP=SHR
//RECON051 DD DSN=IMSVS5.RECON1,DISP=SHR
//RECON052 DD DSN=IMSVS5.RECON2,DISP=SHR
//RECON053 DD DSN=IMSVS5.RECON3,DISP=SHR
//RECON061 DD DSN=IMSVS6.RECON1,DISP=SHR
//RECON062 DD DSN=IMSVS6.RECON2,DISP=SHR
//RECON063 DD DSN=IMSVS6.RECON3,DISP=SHR
//RECON071 DD DSN=IMSVS7.RECON1,DISP=SHR
//RECON072 DD DSN=IMSVS7.RECON2,DISP=SHR
//RECON073 DD DSN=IMSVS7.RECON3,DISP=SHR
//RECON081 DD DSN=IMSVS8.RECON1,DISP=SHR
//RECON082 DD DSN=IMSVS8.RECON2,DISP=SHR
//RECON083 DD DSN=IMSVS8.RECON3,DISP=SHR
//RECON091 DD DSN=IMSVS9.RECON1,DISP=SHR
//RECON092 DD DSN=IMSVS9.RECON2,DISP=SHR
//RECON093 DD DSN=IMSVS9.RECON3,DISP=SHR
//RECON101 DD DSN=IMSVSA.RECON1,DISP=SHR
//RECON102 DD DSN=IMSVSA.RECON2,DISP=SHR
//RECON103 DD DSN=IMSVSA.RECON3,DISP=SHR
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
```

Figure 48. Example for checking the consistency of RECON data sets
Example: Checking the consistency of DBDs

The following JCL example is for checking the consistency of DBD members in DBDLIBs.

The DBD=DBD* keyword specifies that all the DBD members that start with DBD are checked.

```
//stepname EXEC PGM=FABWMCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB01 DD DSN=IMSVS.DBDLIB1,DISP=SHR
//DBDLIB02 DD DSN=IMSVS.DBDLIB2,DISP=SHR
//DBDLIB03 DD DSN=IMSVS.DBDLIB3,DISP=SHR
//DBDLIB04 DD DSN=IMSVS.DBDLIB4,DISP=SHR
//DBDLIB05 DD DSN=IMSVS.DBDLIB5,DISP=SHR
//DBDLIB06 DD DSN=IMSVS.DBDLIB6,DISP=SHR
//DBDLIB07 DD DSN=IMSVS.DBDLIB7,DISP=SHR
//DBDLIB08 DD DSN=IMSVS.DBDLIB8,DISP=SHR
//DBDLIB09 DD DSN=IMSVS.DBDLIB9,DISP=SHR
//DBDLIB10 DD DSN=IMSVS.DBDLIBA,DISP=SHR
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//FABWCTL DD *
  DBD=DBD*
/*
```

Figure 49. Example for checking the consistency of DBDs
Example: Checking the consistency of PSBs
The following JCL example is for checking the consistency of PSB members in PSBLIBs.

The PSB=PSB* keyword specifies that all the PSB members that start with PSB are checked.

```
//stepname EXEC PGM=FABWMCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//PSBLIB01 DD DSN=IMSVS.PSBLIB1,DISP=SHR
//PSBLIB02 DD DSN=IMSVS.PSBLIB2,DISP=SHR
//PSBLIB03 DD DSN=IMSVS.PSBLIB3,DISP=SHR
//PSBLIB04 DD DSN=IMSVS.PSBLIB4,DISP=SHR
//PSBLIB05 DD DSN=IMSVS.PSBLIB5,DISP=SHR
//PSBLIB06 DD DSN=IMSVS.PSBLIB6,DISP=SHR
//PSBLIB07 DD DSN=IMSVS.PSBLIB7,DISP=SHR
//PSBLIB08 DD DSN=IMSVS.PSBLIB8,DISP=SHR
//PSBLIB09 DD DSN=IMSVS.PSBLIB9,DISP=SHR
//PSBLIB10 DD DSN=IMSVS.PSBLIBA,DISP=SHR
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//FABWCTL DD *
  PSB=PSB*
/*
```

Figure 50. Example for checking the consistency of PSBs
Example: Checking the consistency of ACBs
The following JCL example is for checking the consistency of ACB members in ACBLIBs.

The DBD= keywords specify the DBD-type ACB members to check.

```jcl
//stepname EXEC PGM=FABWCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//ACBLIB01 DD DSN=IMSVS.ACBLIB1,DISP=SHR
//ACBLIB02 DD DSN=IMSVS.ACBLIB2,DISP=SHR
//ACBLIB03 DD DSN=IMSVS.ACBLIB3,DISP=SHR
//ACBLIB04 DD DSN=IMSVS.ACBLIB4,DISP=SHR
//ACBLIB05 DD DSN=IMSVS.ACBLIB5,DISP=SHR
//ACBLIB06 DD DSN=IMSVS.ACBLIB6,DISP=SHR
//ACBLIB07 DD DSN=IMSVS.ACBLIB7,DISP=SHR
//ACBLIB08 DD DSN=IMSVS.ACBLIB8,DISP=SHR
//ACBLIB09 DD DSN=IMSVS.ACBLIB9,DISP=SHR
//ACBLIB10 DD DSN=IMSVS.ACBLIBA,DISP=SHR
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//FABWCTL DD *
  DBD=DBD0001
  DBD=DBD0002
/*
```

Figure 51. Example for checking the consistency of ACBs
Example: Comparing the database definitions across multiple sets of RECON data sets

The following JCL example is for generating a RECON Difference report, which reports details about the RECON fields that differ between multiple sets of RECON data sets.

The utility checks the consistency of database definitions in the DB record fields and DBDS record fields.

```
//stepname EXEC PGM=FABWMCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//IMSV11 DD DSN=IMSV11.RESLIB,DISP=SHR
//IMSV12 DD DSN=IMSV12.RESLIB,DISP=SHR
//IMSV13 DD DSN=IMSV13.RESLIB,DISP=SHR
//RECON011 DD DSN=IMSVS1.RECON1,DISP=SHR
//RECON012 DD DSN=IMSVS1.RECON2,DISP=SHR
//RECON013 DD DSN=IMSVS1.RECON3,DISP=SHR
//RECON021 DD DSN=IMSVS2.RECON1,DISP=SHR
//RECON022 DD DSN=IMSVS2.RECON2,DISP=SHR
//RECON023 DD DSN=IMSVS2.RECON3,DISP=SHR
//RECON031 DD DSN=IMSVS3.RECON1,DISP=SHR
//RECON032 DD DSN=IMSVS3.RECON2,DISP=SHR
//RECON033 DD DSN=IMSVS3.RECON3,DISP=SHR
//FABWOUT DD SYSOUT=A
//FABWSUMM DD SYSOUT=A
//FABWRRPT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//FABWCTL DD *
DIFFREP=YES
DBD=DBD001
DBD=DBD002
DBD=DBD003
/*
```

Figure 52. Example for creating a RECON Difference report
Output from the Multiple Resource Checker utility

The output from the Multiple Resource Checker utility includes the FABWOUT data set, the FABWSUMM data set, and the FABWRRT data set.

FABWOUT data set

The FABWOUT data set contains the information about the parameters that were applied to each control statement and any error messages issued by Multiple Resource Checker.

The following figure shows examples of the messages that are generated in the FABWOUT data set.

---

Figure 53. Messages in the FABWOUT data set

FABWSUMM data set

The FABWSUMM data set contains the Resource Check Summary report. This report contains the results of the consistency check.

The FABWSUMM data set must contain fixed-length records of 133 bytes, and a block size of 133 or a multiple of 133.

This report consists of the following two parts:
• The first part provides a list of data sets that were checked.
• The second part provides a matrix table that summarizes whether differences were found.

Subsections:
• “Report field description”
• “Sample report ” on page 192

Report field description

The first part contains information about the input libraries and the DD numbers that are assigned to the libraries by the utility. These DD numbers correspond to the DD numbers used in the second part of the report.

RECON DD NUMBER
A list of the RECON data sets that were checked.

NUM   Sequential numbers that the utility assigned to the sets of RECON data sets.

DDNAME
DBRC RECON data set names that you specified with the RECONxxm DD statements.
IMS VER
Version and release of IMS that is retrieved from the RECON data sets.

MINVERS
The MINVERS value that is retrieved from the RECON data sets.

DB#
The number of databases that are registered in the RECON data sets.

For RECON data sets that were generated by IMS Version 11, NOT SUPPORTED is shown.

DBDLIB DD NUMBER
PSBLIB DD NUMBER
ACBLIB DD NUMBER
A list of the libraries that were checked.

NUM Sequential numbers that the utility assigned to the libraries.

DDNAME
DD names that you specified with the DBDLIBxx, PSBLIBxx, and ACBLIBxx DD statements.

The second part of the report provides a matrix table that shows whether the resources are found in each library and whether they are the same.

DBD RECORDS
PSB RECORDS
Contains a matrix table for the resources that were compared.

DBDNAME
DBD members that were found in the DBDLIBs, ACBLIBs, and RECON data sets.

PSBNAME
PSB members that were found in the PSBLIBs and ACBLIBs.

RESULT
Whether the resources are the same.

SAME All of the definitions are the same across the checked libraries.

DIFF Members in some libraries are different. DIFF is printed when one of the following conditions is met:
  • Definitions in the members are different.
  • The member does not exist in some libraries.
  • The DBD is not registered in the RECON data sets.

When the utility checks for consistency across multiple sets of RECON data sets, only the differences in the DB record fields and the DBDS record fields in the RECON data sets are used to determine the value for the RESULT field. Differences that are found in the RECON record fields and in the data group fields do not affect the value that is shown in the RESULT field.

RECON DD NUMBER
Each number indicates a set of RECON data sets. The numbers are assigned by the utility. See the RECON DD NUMBER field in the first part of the report to identify which DD number corresponds to which set of RECON data sets.
**DBD DD NUMBER**

Each number indicates a DBD library. The numbers are assigned by the utility. See the DBD DD NUMBER field in the first part of the report to identify which DD number corresponds to which DBD library.

**PSB DD NUMBER**

Each number indicates a PSB library. The numbers are assigned by the utility. See the PSB DD NUMBER field in the first part of the report to identify which DD number corresponds to which PSB library.

**ACB DD NUMBER**

Each number indicates an ACB library. The numbers are assigned by the utility. See the ACB DD NUMBER field in the first part of the report to identify which DD number corresponds to which ACB library.

**Asterisk (*)**

Indicates that the member exists in the library. For RECON data sets, it indicates that the DBD is registered in the set of the RECON data sets.

**Hyphen (-)**

Indicates that the library or RECON data sets are not specified in the JCL stream.

**(Blank)**

Indicates that the member does not exist in the library. For RECON data sets, it indicates that the DBD is not registered in the RECON data sets.

**Sample report**

The following figure shows an example of the Resource Check Summary report.
Figure 54. Example of the Resource Check Summary report
**FABWRRPT data set**

The FABWRRPT data set contains the RECON Difference report, which contains names of the RECON fields that were compared.

The FABWRRPT data set must contain fixed-length records of 133 bytes, and a block size of 133 or a multiple of 133.

The Multiple Resource Checker utility checks whether the definitions that relate to the database and the database recovery environment in the following RECON fields are the same:
- RECON record fields
- Data group record fields
- DB record fields
- DBDS record fields

When a difference is found, the utility reports the difference by showing a DIFF indicator in the FIELD column.

**Subsections:**
- "Report field description"
- "Sample report " on page 195

**Report field description**

This report consists of the following fields.

**TYPE**=
**DBNAME=dbname**

**TYPE=** shows the type of the RECON record field. **DBNAME=** is printed before **TYPE=**.

**FIELD (SAME | DIFF)**

The name of the RECON field and an indicator that shows whether the field values are the same.

For information about the RECON record fields, see the topic "Fields in a RECON listing, by record type" in *IMS Commands*. 
The utility does not check all of the fields in RECON data sets. For the fields that are compared, see “Fields compared in RECON data sets” on page 178.

**DD**
Each number indicates a set of RECON data sets. The numbers are assigned by the utility. See the RECON DD NUMBER field in the first part of the Resource Check Summary report to identify which DD number corresponds to which set of RECON data sets. **** indicates that the fields are the same across RECON data sets.

**VALUE**
The value that is defined in each set of RECON data sets.
If the field is not defined in the RECON data sets, N/A is printed.
For RECON data sets that were generated by an IMS version that does not support the field, NOT SUPPORTED is shown.

**Sample report**
The following figure shows an example of the RECON Difference report.
<table>
<thead>
<tr>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORCER</td>
<td>** NOFORCER</td>
<td>LOG DSN CHECK</td>
<td>** CHECK17</td>
<td>STARTNEW</td>
<td>** ND</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>GTF TRACE</td>
<td>** TRACEOFF</td>
<td>SSID</td>
<td>** <strong>NULL</strong></td>
<td>LIST DLOG</td>
<td>** ND</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>CA/IC/LOG CATALOG</td>
<td>** NO</td>
<td>LOGRET PERIOD</td>
<td>** 0.001 0:00:00.0</td>
<td>COMMAND AUTH</td>
<td>** NONE</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>HLO</td>
<td>** <strong>NULL</strong></td>
<td>RCNQUAL</td>
<td>** <strong>NULL</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>CATALOG</td>
<td>** <strong>NULL</strong></td>
<td>ACCESS</td>
<td>** SERIAL</td>
<td>LIST</td>
<td>** STATIC</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>SIZALERT DSNUM</td>
<td>** 15</td>
<td>SIZALERT VOLNUM</td>
<td>** ** 16</td>
<td>SUZALERT PERCENT</td>
<td>** 95</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>LOGALERT DSNUM</td>
<td>** 3</td>
<td>LOGALERT VOLNUM</td>
<td>** 16</td>
<td>REORG# VERIFY</td>
<td>** ND</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td>FIELD (SAME)</td>
<td>DD VALUE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>IMSPLEX</td>
<td>** ** NONE **</td>
<td>GROUP ID</td>
<td>** ** NONE **</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 55. Example of the RECON Difference report when CHKRECON=DBDEF_RCV is specified (Part 1 of 3)
### IMS Library Integrity Utilities - Multiple Resource Checker

#### RECON Difference Report

**Page:** 3  
**Date:** 12/19/2015  
**Time:** 10.47.15  
**Version:** FABWMCHEK - V2.R2

**DBNAME=DB00001 TYPE=DB**

<table>
<thead>
<tr>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DBD</strong></td>
<td><strong>DB00001</strong></td>
<td><strong>SHARE LEVEL</strong></td>
<td><strong>3</strong></td>
<td><strong>DBRCVGRP</strong></td>
<td>**<strong>NULL</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (DIFF)</th>
<th>DD VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RECOVERABLE</strong></td>
<td><strong>NO</strong></td>
<td><strong>DB LEVEL TRACK</strong></td>
<td><strong>N/A</strong></td>
<td><strong>IC NEEDED</strong></td>
<td>01 <strong>DISABLED</strong></td>
</tr>
<tr>
<td>****</td>
<td>****</td>
<td>****</td>
<td>****</td>
<td>****</td>
<td>****</td>
</tr>
</tbody>
</table>

**DBNAME=DB00001 TYPE=DBDS**

<table>
<thead>
<tr>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DDN</strong></td>
<td><strong>DD0000A</strong></td>
<td><strong>DBD</strong></td>
<td><strong>DB00001</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DSN</strong></td>
<td><strong>IMSVS.TEST.IMS12.DD0000A</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (DIFF)</th>
<th>DD VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TYPE</strong></td>
<td><strong>IMS</strong></td>
<td><strong>DBORG</strong></td>
<td><strong>HDAM</strong></td>
<td><strong>DSORG</strong></td>
<td><strong>01 VSAM</strong></td>
</tr>
<tr>
<td>****</td>
<td>****</td>
<td>****</td>
<td>****</td>
<td>****</td>
<td>****</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FIELD (DIFF)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAGRP</strong></td>
<td><strong>01 CAGRDP03</strong></td>
<td><strong>GENMAX</strong></td>
<td><strong>3</strong></td>
<td><strong>REUSE</strong></td>
<td><strong>NOREUSE</strong></td>
</tr>
<tr>
<td><strong>02 CAGRDP03</strong></td>
<td><strong>03 CAGRDP03</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FIELD (DIFF)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RECOVPD</strong></td>
<td><strong>01 31</strong></td>
<td><strong>DEFLTJCL</strong></td>
<td>**<strong>NULL</strong></td>
<td><strong>ICJCL</strong></td>
<td><strong>ICJCL</strong></td>
</tr>
<tr>
<td><strong>02 31</strong></td>
<td><strong>03 365</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
<th>FIELD (SAME)</th>
<th>DD VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OICJCL</strong></td>
<td><strong>OICJCL</strong></td>
<td><strong>RECOVJCL</strong></td>
<td><strong>RECOVJCL</strong></td>
<td><strong>RECVJCL</strong></td>
<td><strong>RECVJCL</strong></td>
</tr>
<tr>
<td>****</td>
<td>****</td>
<td>****</td>
<td>****</td>
<td>****</td>
<td>****</td>
</tr>
</tbody>
</table>

---

**Figure 56. Example of the RECON Difference report when CHKRECON=DBDEF_RCV is specified (Part 2 of 3)**
**Figure 57. Example of the RECON Difference report when CHKRECON=DBDEF_RCV is specified (Part 3 of 3)**

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DD</th>
<th>VALUE</th>
<th>FIELD</th>
<th>DD</th>
<th>VALUE</th>
<th>FIELD</th>
<th>DD</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>**</td>
<td>**DB00002</td>
<td>SHARE LEVEL</td>
<td>**</td>
<td>3</td>
<td>RECOVERABLE</td>
<td>**</td>
<td>YES</td>
</tr>
<tr>
<td>FULLSEG DEFAULT</td>
<td>01</td>
<td>NOT SUPPORTED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBRCVGRP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB000002 TYPE=DB

AREA=DD0002A

DBNAME=DB00002 DBNAME=DB00002

DBNAME=DB00002

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

Area ** DD0002A DBB ** DB00002 SHARE LEVEL ** 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG ** DEDB DSORG ** VSAM CAGRDP 01 GRP1000 02 GRP100 03 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX ** 3 REUSE ** NOREUSE RECOVERPD 01 100 02 100 03 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO ** NOVSO PREOPEN ** NOPREOPEN PRELOAD ** NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA ** DD0002A DBD ** DB00002 SHARE LEVEL ** 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG ** DEDB DSORG VSAM CAGRDP 01 GRP1000 02 GRP100 03 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX ** 3 REUSE ** NOREUSE RECOVERPD 01 100 02 100 03 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO ** NOVSO PREOPEN ** NOPREOPEN PRELOAD ** NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA DD0002A DBD DB00002 SHARE LEVEL 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG DEDB DSORG VSAM CAGRDP GRP1000 GRP100 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX 3 REUSE NOREUSE RECOVERPD 100 100 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO NOVSO PREOPEN NOPREOPEN PRELOAD NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA DD0002A DBD DB00002 SHARE LEVEL 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG DEDB DSORG VSAM CAGRDP GRP1000 GRP100 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX 3 REUSE NOREUSE RECOVERPD 100 100 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO NOVSO PREOPEN NOPREOPEN PRELOAD NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA DD0002A DBD DB00002 SHARE LEVEL 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG DEDB DSORG VSAM CAGRDP GRP1000 GRP100 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX 3 REUSE NOREUSE RECOVERPD 100 100 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO NOVSO PREOPEN NOPREOPEN PRELOAD NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA DD0002A DBD DB00002 SHARE LEVEL 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG DEDB DSORG VSAM CAGRDP GRP1000 GRP100 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX 3 REUSE NOREUSE RECOVERPD 100 100 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO NOVSO PREOPEN NOPREOPEN PRELOAD NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA DD0002A DBD DB00002 SHARE LEVEL 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG DEDB DSORG VSAM CAGRDP GRP1000 GRP100 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX 3 REUSE NOREUSE RECOVERPD 100 100 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO NOVSO PREOPEN NOPREOPEN PRELOAD NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA DD0002A DBD DB00002 SHARE LEVEL 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG DEDB DSORG VSAM CAGRDP GRP1000 GRP100 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX 3 REUSE NOREUSE RECOVERPD 100 100 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO NOVSO PREOPEN NOPREOPEN PRELOAD NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA DD0002A DBD DB00002 SHARE LEVEL 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG DEDB DSORG VSAM CAGRDP GRP1000 GRP100 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX 3 REUSE NOREUSE RECOVERPD 100 100 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO NOVSO PREOPEN NOPREOPEN PRELOAD NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA DD0002A DBD DB00002 SHARE LEVEL 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG DEDB DSORG VSAM CAGRDP GRP1000 GRP100 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX 3 REUSE NOREUSE RECOVERPD 100 100 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO NOVSO PREOPEN NOPREOPEN PRELOAD NDPRELOAD

IMS Library Integrity Utilities - Multiple Resource Checker

"RECON Difference Report"

DBNAME=DB00002 TYPE=DBDS AREA=DD00002A

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

AREA DD0002A DBD DB00002 SHARE LEVEL 3

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

DBORG DEDB DSORG VSAM CAGRDP GRP1000 GRP100 GRP300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

GENMAX 3 REUSE NOREUSE RECOVERPD 100 100 300

Field (SAME) DD VALUE Field (SAME) DD VALUE Field (SAME) DD VALUE

VSO NOVSO PREOPEN NOPREOPEN PRELOAD NDPRELOAD
Chapter 6. DBD/PSB/ACB Compare utility

The DBD/PSB/ACB Compare utility compares two IMS control blocks and reports the differences between the control blocks.

Topics:
- “DBD/PSB/ACB Compare utility overview” on page 200
- “Restrictions and considerations for the DBD/PSB/ACB Compare utility” on page 202
- “Comparing IMS control blocks” on page 203
- “JCL requirements for the DBD/PSB/ACB Compare utility” on page 204
- “Control statements for the DBD/PSB/ACB Compare utility” on page 207
- “JCL examples for the DBD/PSB/ACB Compare utility” on page 223
- “Output from the DBD/PSB/ACB Compare utility” on page 227
DBD/PSB/ACB Compare utility overview

The DBD/PSB/ACB Compare utility reports the differences between two control blocks (DBDs, PSBs, or ACBs), of the same type or different types, that have the same name but reside in different IMS libraries. The utility also reports the differences between two control blocks of the same type that have different names and that reside in the same IMS library or in different IMS libraries. If there are no differences, only activity messages are produced. However, you can optionally generate source-level compare reports even when no difference is found.

Subsections:
- “Function overview”
- “Program structure and job step”
- “Data flow” on page 201

Function overview

The utility provides the following functions:

DBD Compare function
This function compares two control blocks, that is, two DBDs or a DBD and a DBD-type ACB that have the same name but reside in different libraries, and produces a report that shows the differences. Additionally, this function compares two DBDs that have different names and that reside in the same library or in different libraries.

PSB Compare function
This function compares two control blocks, that is, two PSBs or a PSB and a PSB-type ACB that have the same name but reside in different libraries, and produces a report that shows the differences. Additionally, this function compares two PSBs that have different names and that reside in the same library or in different libraries.

ACB Compare function
This function compares two control blocks, that is, two ACBs, a DBD-type ACB and a DBD, or a PSB-type ACB and a PSB that have the same name but reside in different libraries, and produces a report that shows the differences. Additionally, this function compares two ACBs that have different names and that reside in the same library or in different libraries.

Program structure and job step

DBD/PSB/ACB Compare consists of one program, FABLCOMP that controls other load modules and compares two control blocks (DBDs, PSBs or ACBs). This program builds and prints a report that shows the differences between two control blocks of the same type or different types that have the same name but that reside in different libraries. It also builds and prints a report that shows differences between two control blocks, of the same type, that have different names and that reside in the same library or different libraries. If no difference is found, the compare report is not created, and only activity messages are produced. However, if SOURCE and NODIFF parameters are both specified on the REPORT statement, the utility generates source-level compare reports even when no difference is found.

This program uses simple input formats that are specified in the SYSIN data set. All activity and error messages are written in the data set that is defined by the SYSOUT DD statement. If the CTLSTMT parameter is specified for the REPORT
statement, the echo of the SYSIN control statements and selected runtime options are written to the SYSPRINT data set.

**Data flow**

The following figure shows the general data flow for DBD/PSB/ACB Compare (FABLCOMP). Input consists of the SYSIN data set and the DBDLIB, PSBLIB, and ACBLIB data sets. Output consists of the reports and activity log.

---

*Figure 58. Data flow for DBD/PSB/ACB Compare*
Restrictions and considerations for the DBD/PSB/ACB Compare utility

Certain restrictions and considerations apply when you use the DBD/PSB/ACB Compare utility.

The DBD/PSB/ACB Compare utility compares the DBD/PSB fields even if they are not used for some IMS versions and releases.

The DBD/PSB/ACB Compare utility supports only the ACBs generated by IMS of the same version and release when generating block-level compare reports.

When the DBD/PSB/ACB Compare utility compares DBD-type ACBs for a DEDB to generate a source-level compare report, the utility also uses the PSB-type ACB that references the DBD to obtain the DBD VERSION= parameter value. If a problem occurs when reading the PSB-type ACB, the DBD VERSION= parameter value is not compared.

When the DBD/PSB/ACB Compare utility compares a PSB that was generated by IMS Version 2 Release 2, the name of the PCB is not compared even if the name is defined.

When you compare ACBs by using a source-level compare report, the following restrictions apply. These restrictions are the same as the restrictions for the DBD/PSB/ACB Reversal utility.

- Because the DBD/PSB/ACB Compare utility cannot obtain segment name information and database name information of the SOURCE parameter for virtually paired logical relationship, the program does not compare these names.
- Because the DBD/PSB/ACB Compare utility cannot obtain information about the INDICES parameter of the SENSEG statement, the program does not compare the fields of the parameter.
- Because the DBD/PSB/ACB Compare utility cannot obtain label information of the DATASET statement, the program does not compare the label fields.

When you compare ACBs with DBDs, the following restrictions apply:

- The DBD/PSB/ACB Compare utility compares only parameters that exist in ABC libraries. When the utility compares DBD-type ACBs with DBDs, it ignores the parameters that exist only in DBD libraries. For information about parameters that are not contained in ACB libraries, see “Restrictions on the generated control statements” on page 306.
- If the index target segment type that the XDFLD statement specifies is assumed to be the index source segment, the program does not compare the SEGMENT parameter.

The NODIFF option, which generates compare reports even when no difference is found, is supported only for source-level compare reports.

For the restrictions that apply to the generated control statements, see “Restrictions on the generated control statements” on page 306.
Comparing IMS control blocks

To compare DBDs, PSBs, and ACBs by using the DBD/PSB/ACB Compare utility, you must prepare the JCL for the DBD/PSB/ACB Compare utility, submit the job, and check the differences in the compare reports.

About this task

Sample JCL for the DBD/PSB/ACB Compare utility is in the SHPSJCL0 library, member FABLIVP1. You can modify this sample JCL and then use it to run the utility.

Procedure

1. In the DBD/PSB/ACB Compare JCL, code the EXEC statement and DD statements.
   See “JCL requirements for the DBD/PSB/ACB Compare utility” on page 204.
2. In the SYSIN data set, code the control statements for the DBD/PSB/ACB Compare utility.
   See “Control statements for the DBD/PSB/ACB Compare utility” on page 207.
3. Submit the job.
4. Check the compare reports that are generated in the output data sets.
   See “Output from the DBD/PSB/ACB Compare utility” on page 227.

What to do next

If you identify differences between two control blocks after running the DBD/PSB/ACB Compare utility, you can run the DBD/PSB/ACB Mapper utility, or the DBD/PSB/ACB Reversal utility to obtain more information about the control blocks.
JCL requirements for the DBD/PSB/ACB Compare utility

When you code the JCL to run the DBD/PSB/ACB Compare utility, include the EXEC statement and appropriate DD statements.

Subsections:
- "JCL example"
- "EXEC statement"
- "DD statements"

JCL example

An example of the JCL that is required for DBD/PSB/ACB Compare is shown in the following figure.

```
//stepname EXEC PGM=FABLCOMP,REGION=512K
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//DBDLIB2 DD DSN=IMSVS.TEST.DBDLIB,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//PSBLIB2 DD DSN=IMSVS.TEST.PSBLIB,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//ACBLIB2 DD DSN=IMSVS.TEST.ACBLIB,DISP=SHR
//SYSSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSSIN DD *
/* (control statements) */
```

Figure 59. Example of DBD/PSB/ACB Compare JCL (FABLCOMP JCL)

EXEC statement

The EXEC statement must be in the following format:

```
//stepname EXEC PGM=FABLCOMP
```

DD statements

Code the following DD statements to identify the source of input and the placement of output information:

**STEPLIB DD or JOBLIB DD**

This DD statement is required. This input DD statement defines the IMS Library Integrity Utilities load module library.

**DBDLIB DD**

This DD statement is required when the DBD= control statement is specified. The DBDLIB DD input data set is the library that contains the DBDs to be compared.

**DBDLIB2 DD**

This DD statement is required when you want to compare a DBD to another DBD, or a DBD-type ACB to a DBD. The DBDLIB2 DD input data set is the library that contains the DBDs to be compared.
Note: When comparing two DBDs that have the same name, the libraries specified in the DBDLIB DD and DBDLIB2 DD statements must be different libraries, and each of them must contain at least one DBD that has the same name.

PSBLIB DD
This DD statement is required when the PSB= control statement is specified. The PSBLIB DD input data set is the library that contains the PSBs to be compared.

PSBLIB2 DD
This DD statement is required when you want to compare a PSB to another PSB, or a PSB-type ACB to a PSB. The PSBLIB2 DD input data set is the library that contains the PSBs to be compared.

Note: When comparing two PSBs that have the same name, the libraries specified in the PSBLIB DD and PSBLIB2 DD statements must be different libraries, and each of them must contain at least one PSB that has the same name.

ACBLIB DD
This DD statement is required when the ACB= control statement is specified. The ACBLIB DD input data set is the library that contains the ACBs (PSB-type ACBs or DBD-type ACB) to be compared.

ACBLIB2 DD
This DD statement is required when you want to compare an ACB (PSB-type ACB or DBD-type ACB) to another ACB, a DBD to a DBD-type ACB, or a PSB to a PSB-type ACB. The ACBLIB2 DD input data set is the library that contains the ACBs (PSB-type ACBs or DBD-type ACB) to be compared.

Note: When comparing two ACBs that have the same name, the libraries specified in the ACBLIB DD and ACBLIB2 DD statements must be different libraries, and each must contain at least one ACB that has the same name.

SYSOUT DD
This DD statement is required. This output data set contains all activity messages and error messages. The record format is fixed-blocked. The logical record length is 133. Block size, if coded, must be a multiple of 133.

SYSPRINT DD
This DD statement is required. This output data set contains the reports of the comparisons made by DBD/PSB/ACB Compare. The reports are classified as DBD, PSB, and ACB Compare reports, and then each group is sorted alphabetically by member name in the DBDLIB, the PSBLIB, and the ACBLIB libraries. This output data set also contains the echo of the SYSIN control statements and selected runtime options when the CTLSTMT parameter is specified for the REPORT statement. The record format is fixed-blocked. The logical record length is 133. Block size, if coded, must be a multiple of 133.

SYSIN DD
This DD statement is required. SYSIN DD is the control data set for this program.

The record format is fixed-blocked. The logical record length is 80. Block size, if coded, must be a multiple of 80.
Up to 9999 control statements can be specified by use of the SYSIN DD statement. If there are more than 9999 control statements, the excess control statements are ignored.

**Related reading:** For the format of the control statements, see "Control statements for the DBD/PSB/ACB Compare utility" on page 207.
Control statements for the DBD/PSB/ACB Compare utility

The input to the DBD/PSB/ACB Compare utility consists of control statements in the SYSIN data set. These control statements contain keywords that indicate the function and the names of the DBDs, PSBs, or ACBs for which the reports are created.

This data set usually resides in the input stream. However, it can be defined as a sequential data set or as a member of a partitioned data set. It must contain one 80-byte fixed-length record for each DBD, PSB, or ACB member to be compared. Block size, if coded, must be a multiple of 80.

Output reports are always generated in the order of DBD Compare reports, PSB Compare reports, and ACB Compare reports, with members in each group sorted alphabetically.

DBD, PSB, ACB control statements
A DBD control statement, PSB control statement, or ACB control statement specifies the member to compare. See “DBD, PSB, ACB control statements” on page 209.

REPORT control statement
A REPORT control statement controls report output. See “REPORT control statement” on page 212.

NOCOMP control statement
A NOCOMP control statement specifies the field that you want to exclude from comparison. See “NOCOMP control statement” on page 213.

Control statement example
The SYSIN data set can be coded as shown in the following figure.

```
//SYSIN DD *
REPORT=SOURCE,NODIFF
NOCOMP=parameter,parameter
DBD=member
DBD=member1:member2
DBD=member,ACB
PSB=member
PSB=member1:member2
PSB=member,ACB
ACB=member
ACB=member1:member2
ACB=member,DBD
ACB=member,PSB
ACB=member,BOTH
/*
```

Figure 60. Examples of control statements for DBD/PSB/ACB Compare

Syntax rules

The control statements for DBD/PSB/ACB Compare must adhere to the following syntax rules:

- Control statements can be coded anywhere between columns 2 - 80.
• In the control statement field, keyword, equal sign, and member name must not be separated by blanks. Because a blank serves as the delimiter, only a comment can be written after a blank.

• A statement with an asterisk (*) in column 1 is treated as a comment.

• The control statements can be specified in any order. For example, in the following order:

```
ACB=XXXXXXXX
PSB=XXXXXXXX
PSB=XXXXXXXX
NOCOMP=XXXXXXXX
ACB=XXXXXXXX
DBD=XXXXXXXX
```
DBD, PSB, ACB control statements

A DBD control statement, PSB control statement, or ACB control statement specifies the members to compare.

Subsections:
- "DBD control statement"
- "PSB control statement"
- "ACB control statement"
- "Use of wildcards " on page 210
- "Quick reference for DBD, PSB, ACB control statements and DD statements” on page 210

DBD control statement

DBD=member
The members (member) in the libraries specified in the DBDLIB DD and DBDLIB2 DD statements are to be compared. The result is written to DBD Compare reports.

DBD=member1:member2
The member (member1) in the library that is specified in the DBDLIB DD statement and the member (member2) in the library that is specified in the DBDLIB2 DD statement are to be compared. The results are written to the DBD Compare report.

DBD=member,ACB
The members (member) in the libraries that are specified in the DBDLIB DD and ACBLIB2 DD statements are to be compared at their source level. The result is written to DBD Compare reports. The abbreviation A can be used instead of the parameter ACB.

PSB control statement

PSB=member
The members (member) in the libraries specified in the PSBLIB DD and PSBLIB2 DD statements are to be compared. The result is written to PSB Compare reports.

PSB=member1:member2
The member (member1) in the library that is specified in the PSBLIB DD statement and the member (member2) in the library that is specified in the PSBLIB2 DD statement are to be compared. The results are written to the PSB Compare report.

PSB=member,ACB
The members (member) in the libraries that are specified in the PSBLIB DD and ACBLIB2 DD statements are to be compared at their source level. The result is written to PSB Compare reports. The abbreviation A can be used instead of the parameter ACB.

ACB control statement

ACB=member
The members (member) in the libraries specified in the ACBLIB DD and ACBLIB2 DD statements are to be compared. The result is written to ACB Compare reports.
ACB=\textit{member1:member2}

The member (\textit{member1}) in the library that is specified in the ACBLIB DD statement and the member (\textit{member2}) in the library that is specified in the ACBLIB2 DD statement are to be compared. The results are written to the ACB Compare report.

ACB=\textit{member,parameter}

The members (\textit{member}) in the libraries that are specified in the ACBLIB DD statement are to be compared to the members in the libraries that are specified in the DBDLIB2 DD, PSBLIB2 DD, or in both DD statements at their source level. The results are written to ACB Compare reports.

The following options can be specified for \textit{parameter} (the allowed abbreviation is shown in parentheses):

\textbf{DBD (D)}

This option specifies that the members in the libraries that are specified in the ACBLIB are to be compared to the members in the libraries that are specified in DBDLIB2 at their source level.

\textbf{PSB (P)}

This option specifies that the members in the libraries that are specified in the ACBLIB are to be compared to the members in the libraries that are specified in PSBLIB2 at their source level.

\textbf{BOTH (B)}

This option specifies that the members in the libraries that are specified in the ACBLIB are to be compared to the members in the libraries specified in DBDLIB2 and PSBLIB2 at their source level.

\section*{Use of wildcards}

For \textit{member} and \textit{member1} (the first member name), you can specify a wildcard in any position of a character string. The asterisk (*) and the percent sign (%) are supported as wildcard characters. An asterisk represents 0 - 8 characters, and a percent sign represents a single character. If two or more asterisks are specified sequentially, only the first asterisk is recognized. You cannot use wildcard characters to specify \textit{member2} (the second member name).

\section*{Quick reference for DBD, PSB, ACB control statements and DD statements}

The following table lists the DBD/PSB/ACB Compare functions, control statements, and DD statements.

\begin{table}[h]
\centering
\caption{DBD/PSB/ACB Compare functions, control statements, and DD statements}
\begin{tabular}{|c|c|c|c|c|c|c|c|c|c|}
\hline
\textbf{Function} & \textbf{Control keyword} & \textbf{Required DD statements} & \textbf{SYS PRINT} & \textbf{SYS OUT} & \textbf{DBD LIB} & \textbf{DBD LIB2} & \textbf{PSB LIB} & \textbf{PSB LIB2} & \textbf{ACB LIB} & \textbf{ACB LIB2} & \textbf{SYS IN} \\
\hline
DBD compare & DBD= & & R & R & R & R & & & & R & \\
 & DBD=\textit{member1:member2} & & R & R & R & R & & & & & \\
 & DBD=\textit{member,A} & & R & R & & R & & & & & \\
PSB compare & PSB= & & R & R & & & & & & & \\
 & PSB=\textit{member1:member2} & & R & R & & & & & & & \\
 & PSB=\textit{member,A} & & R & R & & & & & & & \\
\hline
\end{tabular}
\end{table}
<table>
<thead>
<tr>
<th>Function</th>
<th>Control keyword</th>
<th>Required DD statements (O: Optional R: Required)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SYS PRINT</td>
</tr>
<tr>
<td>ACB compare</td>
<td>ACB=</td>
<td></td>
</tr>
<tr>
<td>ACB=member1:member2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACB=member,D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACB=member,P</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACB=member,B</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
REPORT control statement

A REPORT control statement controls report output. The REPORT statement applies to all members that are specified in the SYSIN DD statement, regardless of the order of the statements.

The REPORT control statement supports the following parameters:

SOURCE
This parameter specifies to generate the source-level compare reports in the SYSPRINT data set.

NODIFF
This parameter specifies to generate the source-level compare reports even when no difference is found. This parameter is effective only when the SOURCE parameter is specified on the REPORT statement.

CTLSTMT
This parameter specifies to generate the control statement report in the SYSPRINT data set.

Examples

You can code the REPORT control statement in the following ways:

- REPORT=SOURCE
- REPORT=CTLSTMT
- REPORT=SOURCE,NODIFF
- REPORT=SOURCE,NODIFF,CTLSTMT
- REPORT=SOURCE
  REPORT=NODIFF
  REPORT=CTLSTMT
NOCOMP control statement

A NOCOMP control statement specifies the field that you want to exclude from comparison. A NOCOMP control statement applies to all members that the SYSIN DD statement specifies, regardless of the order of the statements.

Subsections:
- “Summary of NOCOMP keyword parameters for source-level compare”
- “Summary of NOCOMP keyword parameters for block-level compare” on page 214
- “Description of NOCOMP keyword parameters” on page 219
- “Examples” on page 222

Summary of NOCOMP keyword parameters for source-level compare

The following table summarizes the NOCOMP keyword parameters and, for each parameter, the statements and parameters that are not compared. Refer to this table when you compare members at the source level.

Table 14. Source-level compare: Statements and parameters that are not compared

<table>
<thead>
<tr>
<th>NOCOMP keyword parameter</th>
<th>Affected member type</th>
<th>Statements and parameters not compared</th>
</tr>
</thead>
<tbody>
<tr>
<td>AREA</td>
<td>DBD</td>
<td>AREA statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AREA statements and any parameters on the AREA statements</td>
</tr>
<tr>
<td>COMPRTN</td>
<td>DBD</td>
<td>SEGMENT statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COMPRTN=</td>
</tr>
<tr>
<td>DBDNAME</td>
<td>DBD</td>
<td>DBD statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NAME=</td>
</tr>
<tr>
<td>IMSREL</td>
<td>DBD</td>
<td>AREA, DATASET statements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DEVICE= (Removed by IMS Version 4. If a member is generated by IMS V3 or earlier and the other member is generated by IMS V4 or later, the utility does not compare this parameter.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DBD statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATXEXIT= (Added by IMS Version 3. If a member is generated by IMS V2 or earlier and the other is generated by IMS V3 or later, the utility does not compare this parameter.)</td>
</tr>
<tr>
<td>PSB</td>
<td>PCB</td>
<td>PCB statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PCBNAME=, LIST= (Added by IMS Version 3. If a member is generated by IMS V2 or earlier and the other is generated by IMS V3 or later, the utility does not compare these parameters.)</td>
</tr>
<tr>
<td>KEYLEN</td>
<td>PSB</td>
<td>PCB statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KEYLEN=</td>
</tr>
<tr>
<td>LANG</td>
<td>PSB</td>
<td>PSBGEN statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANG=</td>
</tr>
<tr>
<td>LIST</td>
<td>PSB</td>
<td>PCB statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LIST=</td>
</tr>
</tbody>
</table>
### Table 14. Source-level compare: Statements and parameters that are not compared (continued)

<table>
<thead>
<tr>
<th>NOCOMP keyword parameter</th>
<th>Affected member type</th>
<th>Statements and parameters not compared</th>
</tr>
</thead>
<tbody>
<tr>
<td>METADATA (or CATALOG)</td>
<td>DBD</td>
<td>DFSMARSH, DFSMAP, DFSCASE statements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>These statements and any parameters on these statements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FIELD statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CASENAME=, DATATYPE=, DEPENDSON=, EXTERNALNAME=,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINOCCURS=, MAXOCCURS=, MAXBYTES=, PARENT=,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINES=, RELSTART=, REMARKS=, STARTAFTER=</td>
</tr>
<tr>
<td></td>
<td>PSB</td>
<td>ENCODING=, EXTERNALNAME=, REMARKS=</td>
</tr>
<tr>
<td>PCBNAME</td>
<td>PSB</td>
<td>PCB statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PCBNAME= or label</td>
</tr>
<tr>
<td>PROCOPT</td>
<td>PSB</td>
<td>DB PCB, GSAM PCB, SENSEG statements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PROCOPT=</td>
</tr>
<tr>
<td>PROCSEQ</td>
<td>PSB</td>
<td>DB PCB statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PROCSEQ=</td>
</tr>
<tr>
<td>PROCSEQD</td>
<td>PSB</td>
<td>DB PCB statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PROCSEQD=</td>
</tr>
<tr>
<td>PSB_ACCESS</td>
<td>PSB</td>
<td>DB PCB statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ACCESS=</td>
</tr>
<tr>
<td>PSB_PSELOPT</td>
<td>PSB</td>
<td>DB PCB statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSELOPT=</td>
</tr>
<tr>
<td>PSBNAME</td>
<td>PSB</td>
<td>PSBGEN statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSBNAME=</td>
</tr>
<tr>
<td>RMNAME</td>
<td>DBD</td>
<td>DBD statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RMNAME=</td>
</tr>
<tr>
<td>VERSION</td>
<td>DBD</td>
<td>DBD statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VERSION=</td>
</tr>
</tbody>
</table>

#### Summary of NOCOMP keyword parameters for block-level compare

The following table summarizes the NOCOMP keyword parameters and, for each parameter, the fields that are not compared. Refer to this table when you compare members at the block level.
<table>
<thead>
<tr>
<th>NOCOMP keyword parameter</th>
<th>Affected member type</th>
<th>Fields that are not compared (Section and field description)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AREA</td>
<td>DBD</td>
<td>Fields related to the AREA statement and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS IDBD macro - PRFX DB section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NO OF SEGMENTS</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS IDBD macro - PRFX DSG section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LOGICAL RECORD LENGTH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• OVERFLOW/OUTPUT LOGICAL RECORD LENGTH</td>
</tr>
<tr>
<td>DBD-type ACB</td>
<td></td>
<td>Fields related to the AREA statement and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DBFDMCB macro - DMCB DBD section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDRESS OF FDT ENTRY FOR ROOT KEY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MAXIMUM IOA LENGTH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• OFFSET FROM BEGINNING OF DMCB</td>
</tr>
<tr>
<td>COMPRTN</td>
<td>DBD</td>
<td>Fields related to the COMPRTN= parameter of the SEGM statement and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS IDBD macro - PRFX DSG section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LONGEST SEGMENT LENGTH</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS IDBD macro - SEGTAB SEG section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MIN LENGTH OR ZERO FOR FIX LENGTH SEGS</td>
</tr>
<tr>
<td>DBD-type ACB</td>
<td></td>
<td>Fields related to the COMPRTN= parameter of the SEGM statement and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSDMB macro - PRFX section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DMB SIZE IN BYTES</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSDMB macro - AMPBPX section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LENGTH OF LARGEST SEGMENT IN DATASET</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSDMB macro - PSDB section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FOR VAR LENGTH SEG - MIN VALUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DBFDMCB macro - DMCB DBD section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDRESS OF FDT ENTRY FOR ROOT KEY.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• OFFSET FROM BEGINNING OF DMCB</td>
</tr>
<tr>
<td>DBDNAME</td>
<td>DBD, DBD-type ACB</td>
<td>Fields related to the DBD NAME= statement</td>
</tr>
<tr>
<td>DMBNUM</td>
<td>DBD-type ACB</td>
<td>DMB number field</td>
</tr>
<tr>
<td>NOCOMP keyword parameter</td>
<td>Affected member type</td>
<td>Fields that are not compared (Section and field description)</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------</td>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td>IMSREL</td>
<td>DBD</td>
<td><strong>IMS IDBD macro</strong> - <strong>DIR section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DBDGEN DONE ON IMSV12 OR LATER (Added by IMS Version 12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS IDBD macro</strong> - <strong>PRFX DB section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NO OF AREAS(NEW) (Added by IMS Version 8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this field exists in both members, the utility compares this field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS IDBD macro</strong> - <strong>PRFX DSG section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AREA ID(NEW) (Added by IMS Version 8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this field exists in both members, the utility compares this field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DEVICE TYPE OR RESERVED FIELD (Removed by IMS Version 4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS IDBD macro</strong> - <strong>DBDXTB section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;DBDX&quot; EYECATCHER (Added by IMS Version 3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LENGTH OF DBDXTAB (Added by IMS Version 3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LEVEL OF DBDGEN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CALL DFSDBUX1 USER EXIT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;V&quot; = VERSION ID,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;T&quot; = TIMESTAMP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LENGTH OF VERSION ID (HEX)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VERSION ID (VARIABLE LENGTH)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS IDBD macro</strong> - <strong>EXITTB section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NODLET OPTION (Y/N)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NO BEFORE OPTION (Y/N)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DBDGEN macro</strong> - <strong>DBDGEN section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IMS RELEASE LEVEL</td>
</tr>
<tr>
<td>PSB</td>
<td></td>
<td><strong>IMS DFSIPSB macro</strong> - <strong>PRFX section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• INCREASED LIMIT OF SENSEGS (Added by IMS Version 7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IMS V12 FLAG (Added by IMS Version 12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro</strong> - <strong>SENSEG section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PARENT OFFSET IN SEGTBL (Added by IMS Version 7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS PSBGEN macro</strong> - <strong>PSBGEN section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IMS RELEASE LEVEL (Added by IMS Version 3)</td>
</tr>
<tr>
<td>NOCOMP keyword parameter</td>
<td>Affected member type</td>
<td>Fields that are not compared (Section and field description)</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------</td>
<td>----------------------------------------------------------</td>
</tr>
<tr>
<td>KEYLEN</td>
<td>PSB</td>
<td>Fields related to the KEYLEN= parameter of the PCB statement</td>
</tr>
<tr>
<td></td>
<td>PSB-type ACB</td>
<td>Fields related to the KEYLEN= parameter of the PCB statement and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro - PRFX section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE (W/O INTENT LIST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE (WITH INTENT LIST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SIZE OF CSA PART OF PSB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOTAL SIZE OF PSB WORK AREA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MAX I/O WORK AREA SIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDR OF FAST PATH CONTROL BLOCK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LENGTH OF THIS DBPCB MINUS PREFIX</td>
</tr>
<tr>
<td>LANG</td>
<td>PSB, PSB-type ACB</td>
<td>Field related to the LANG= parameter</td>
</tr>
<tr>
<td>LIST</td>
<td>PSB, PSB-type ACB</td>
<td>Field related to the LIST= parameter</td>
</tr>
<tr>
<td>METADATA (or CATALOG)</td>
<td>DBD, DBD-type ACB</td>
<td>Fields related to the metadata fields in DBD, PSB, or ACB</td>
</tr>
<tr>
<td>PCBNAME</td>
<td>PSB</td>
<td>Fields related to the PCBNAME= parameter, the label parameter of the PCB statement, and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro - DBPCB PCB section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SEGMENT NAME FEEDBACK</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro - GSPCB PCB section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SEGMENT NAME FEEDBACK</td>
</tr>
<tr>
<td></td>
<td>PSB-type ACB</td>
<td>Fields related to the PCBNAME= parameter, the label parameter of the PCB statement, and the following field:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro - DBPCB PCB section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SEGMENT NAME FEEDBACK</td>
</tr>
</tbody>
</table>
Table 15. Block-level compare: Fields that are not compared (continued)

<table>
<thead>
<tr>
<th>NOCOMP keyword parameter</th>
<th>Affected member type</th>
<th>Fields that are not compared (Section and field description)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCOPT</td>
<td>PSB</td>
<td>Fields related to the PROCOPT= parameter</td>
</tr>
<tr>
<td></td>
<td>PSB-type ACB</td>
<td>Fields related to the PROCOPT= parameter and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro - PPFX section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SIZE OF INDEX MAINT WORK AREA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SIZE OF SEGMENT WORK AREA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SIZE NEEDED FOR UPDATED DB LIST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SIZE OF SEGWK FOR GO EXPANSION</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE (W/O INTENT LIST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE (WITH INTENT LIST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOTAL SIZE OF PSB WORK AREA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MAX I/O WORK AREA SIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDR OF FAST PATH CONTROL BLOCK</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSSDBM macro - SDB SEG (segname) section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RELATIVE OFFSET TO THE PHYSICAL DSG OF THIS SDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDR OF THE DSG SECTION OF THE JCB FOR THIS SEG</td>
</tr>
<tr>
<td>PROCSEQ</td>
<td>PSB</td>
<td>Fields related to the PROCSEQ= parameter</td>
</tr>
<tr>
<td></td>
<td>PSB-type ACB</td>
<td>Fields related to the PROCSEQ= parameter and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro - PRFX section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SIZE OF INDEX MAINT WORK AREA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE (W/O INTENT LIST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE (WITH INTENT LIST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOTAL SIZE OF PSB WORK AREA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MAX I/O WORK AREA SIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDR OF FAST PATH CONTROL BLOCK</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSSDBM macro - SDB SEG (segname) section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SECONDARY INDEX IS MAIN PROCESSING SEQ.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RELATIVE OFFSET TO THE PHYSICAL DSG OF THIS SDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDR OF THE DSG SECTION OF THE JCB FOR THIS SEG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SDB LOGICALLY RELATED</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDRESS OF SDB EXPANSION</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SEGMENT IS RETRIEVED VIA INDEX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SDB EXPANSION FOR SECONDAR IND</td>
</tr>
</tbody>
</table>
Table 15. Block-level compare: Fields that are not compared  (continued)

<table>
<thead>
<tr>
<th>NOCOMP keyword parameter</th>
<th>Affected member type</th>
<th>Fields that are not compared (Section and field description)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCSEQD</td>
<td>PSB</td>
<td>Fields related to the PROCSEQD= parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro - PRFX section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PST ADDR OF SCHED REGIN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EITHER 0000 OR THE OFFSET FROM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE</td>
</tr>
<tr>
<td></td>
<td>PSB-type ACB</td>
<td>Fields related to the PROCSEQD= parameter and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro - PRFX section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MAXIMUM DBPCB KEY FEEDBACK LENGTH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NUMBER OF DBPCBS IN THIS PSB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOTAL NUMBER OF PCBS IN THIS PSB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE (W/O INTENT LIST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE (WITH INTENT LIST)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SIZE OF CSA PART OF PSB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDR OF USER PARMS IN CTRL RGN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MAX I/O WORK AREA SIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ADDR OF FAST PATH CONTROL BLOCK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IMS/VS DL/I DB ACCESS INDICATOR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• OFFSET TO THE FIRST GSAM PCB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSB SIZE</td>
</tr>
<tr>
<td>PSB_ACCESS</td>
<td>PSB, PSB-type ACB</td>
<td>Fields related to the ACCESS= parameter</td>
</tr>
<tr>
<td>PSB_PSELOPT</td>
<td>PSB, PSB-type ACB</td>
<td>Fields related to the PSELOPT= parameter</td>
</tr>
<tr>
<td>PSBNAME</td>
<td>PSB, PSB-type ACB</td>
<td>Fields related to the PSBNAME= parameter</td>
</tr>
<tr>
<td>RMNAME</td>
<td>DBD</td>
<td>Fields related to the RMNAME= parameter</td>
</tr>
<tr>
<td></td>
<td>DBD-type ACB</td>
<td>Fields related to the RMNAME= parameter and the following fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMS DFSIPSB macro - PRFX section</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DMB SIZE IN BYTES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ECB FOR BACKGROUND WRITE TO POST</td>
</tr>
<tr>
<td>VERSION</td>
<td>DBD, DBD-type ACB</td>
<td>Fields related to the VERSION= parameter</td>
</tr>
</tbody>
</table>

**Description of NOCOMP keyword parameters**

With the following parameters, this statement specifies which field in all of the members specified in the SYSIN DD statement is not compared:
AREA  This parameter indicates that for DBD and DBD-type ACB members, the AREA statements of the DBDGEN utility and the fields that relate to the AREA statements are not compared.

The AREA statement of the DBDGEN utility defines an area within the database, and it also affects fields in the member. If NOCOMP=AREA is specified, the fields that are defined by the AREA statement and any fields affected by the AREA statement are not compared.

COMPRTN
This parameter indicates that for DBD and DBD-type ACB members, the fields that relate to the COMPRTN= parameter of the SEGM statement of the DBDGEN utility are not compared.

The COMPRTN= parameter of the DBDGEN utility defines the segment compression parameters, and it also affects fields in the member. If NOCOMP=COMPRTN is specified, the fields that are defined by the COMPRTN= parameter and the fields affected by the COMPRTN= parameter are not compared.

DBDNAME
This parameter indicates that the fields that are related to the DBD NAME= statement, which was used for creating the DBD or DBD-type ACB member, are not compared. DBD names that are specified by other statements, such as external DBD names, are compared even if this parameter is specified. This parameter is useful for comparing members that have different names.

DMBNUM
This parameter indicates that the DMB number field in DBD-type ACB is not compared. This field value depends only on the order of ACB generations.

IMSREL
Even if two members are generated from the same source, if the version of IMS that generated the two members is different, the compare utility might report some differences between the members. If you specify NOCOMP=IMSREL, the utility does not compare such difference and reports that the two members are the same.

NOCOMP=IMSREL works as follows:

• For source-level compare, the utility does not compare the parameters that were added or removed by certain IMS versions.
• For block-level compare, the utility does not compare the fields that are different between certain IMS versions or that were added or removed by certain IMS versions.

Usage note: Whether NOCOMP=IMSREL is specified or not, the utility does not compare the IMS release that generated the DBD or the PSB members.

KEYLEN
This parameter indicates that for PSB and PSB-type ACB members, the fields that relate to the KEYLEN= parameter of the PCB statement of the PSBGEN utility are not compared.

The KEYLEN= parameter of the PSBGEN utility defines the length of the longest concatenated key for the PCB, and it also affects fields in the
member. If NOCOMP=KEYLEN is specified, the fields that are defined by the KEYLEN= parameter and the fields affected by the KEYLEN= parameter are not compared.

LANG
This parameter indicates that for PSB and PSB-type ACB members, the field that relates to the LANG= parameter of the PSBGEN utility is not compared.

LIST
This parameter indicates that for PSB and PSB-type ACB members, the field that relates to the LIST= parameter of the PSBGEN utility is not compared.

METADATA
This parameter indicates that the metadata fields in DBD, PSB, or ACB are not compared.

Instead of specifying NOCOMP=METADATA, you can specify NOCOMP=CATALOG. CATALOG is an alias for METADATA.

PCBNAME
This parameter indicates that for PSB and PSB-type ACB members, the fields that relate to the PCBNAME= parameter and the label parameter of the PCB statement of the PSBGEN utility are not compared.

The PCBNAME= parameter and the label parameter of the PSBGEN utility define the name of the PCB, and they also affect fields in the member. If NOCOMP=PCBNAME is specified, the fields that are defined by these parameters and the fields affected by the parameters are not compared.

PROCOPT
This parameter indicates that for PSB and PSB-type ACB members, the fields that relate to the PROCOPT= parameter of the PSBGEN utility are not compared.

The PROCOPT= parameter of the PSBGEN utility defines the processing options, and it also affects fields in the member. If NOCOMP=PROCOPT is specified, the fields that are defined by the PROCOPT= parameter and the fields affected by the PROCOPT= parameter are not compared.

PROCSEQ
This parameter indicates that for PSB and PSB-type ACB members, the fields that relate to the PROCSEQ= parameter of the PSBGEN utility are not compared.

The PROCSEQ= parameter of PSBGEN utility defines the name of a secondary index, and it also affects fields in the member. If NOCOMP=PROCSEQ is specified, the fields that are defined by the PROCSEQ= parameter and the fields affected by the PROCSEQ= parameter are not compared.

PROCSEQD
This parameter indicates that for PSB and PSB-type ACB members, the fields that relate to the PROCSEQD= parameter of the PSBGEN utility are not compared.

The PROCSEQD= parameter of the PSBGEN utility defines the name of a secondary index for the primary DEDB database, and it also affects fields in the member. If NOCOMP=PROCSEQD is specified, the fields that are defined by the PROCSEQD= parameter and the fields affected by the PROCSEQD= parameter are not compared.
PSB_ACCESS
This parameter indicates that for PSB and PSB-type ACB members, the field that relates to the ACCESS= parameter of the PSBGEN utility is not compared.

PSB_PSELOPT
This parameter indicates that for PSB and PSB-type ACB members, the field that relates to the PSELOPT= parameter of the PSBGEN utility is not compared.

PSBNAME
This parameter indicates that the fields that are related to the PSBNAME= parameter, which was used for creating the PSB or PSB-type ACB member, are not compared. This parameter is useful for comparing members that have different names.

RMNAME
This parameter indicates that for DBD and DBD-type ACB members, the fields that relate to the RMNAME= parameter of the DBD statement of the DBDGEN utility are not compared.

The RMNAME= parameter of the DBDGEN utility defines the randomizing parameters, and it also affects fields in the member. If NOCOMP=RMNAME is specified, the fields that are defined by the RMNAME= parameter and the fields affected by the RMNAME= parameter are not compared.

VERSION
This parameter indicates that for DBD and DBD-type ACB members, the fields related to the VERSION= parameter of the DBDGEN utility is not compared.

Examples

You can specify the NOCOMP control statement in one of the following formats:

- To specify a single parameter, code the statement as follows:
  NOCOMP=VERSION

- To specify multiple parameters, use commas to separate the parameters. For example:
  NOCOMP=VERSION,DBDNAME,AREA

- To specify multiple parameters that cannot fit on one line, code multiple NOCOMP keywords as follows:
  NOCOMP=VERSION,DMBNUM,IMSREL
  NOCOMP=DBDNAME,PSBNAME
  NOCOMP=AREA,RMNAME,COMPRTN,PCBNAME,KEYLEN
  NOCOMP=LANG,LIST,PROCLOPT,PROCSQ
  NOCOMP=PROCSQD,PSB_PSELOPT,PSB_ACCESS
  NOCOMP=METADATA
JCL examples for the DBD/PSB/ACB Compare utility

The following topics provide JCL examples for running the DBD/PSB/ACB Compare utility to compare DBDs, PSBs, and ACBs.

Example: Comparing two DBDs

The following figure shows example JCL for running the DBD/PSB/ACB Compare utility to compare several DBDs in DBD libraries IMSVS.DBDLIB and IMSVS.TEST.DBDLIB.

```
//stepname EXEC PGM=FABLCOMP
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//DBDLIB2 DD DSN=IMSVS.TEST.DBDLIB,DISP=SHR
//SYSOOT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
DBD=BE1PARTS
DBD=BE2LORDR
DBD=BE2ORDER
DBD=BE2PARTS
/*
```

Figure 61. Example of creating a DBD Compare report—Comparing two DBDs

Example: Comparing two DBDs that have different names

The following figure shows example JCL for comparing DBDs that have different member names. The members to be compared are delimited by a colon. In this example, the members before the colon are in DBD library IMSVS.DBDLIB, and the members after the colon are in DBD library IMSVS.TEST.DBDLIB.

```
//stepname EXEC PGM=FABLCOMP
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//DBDLIB2 DD DSN=IMSVS.TEST.DBDLIB,DISP=SHR
//SYSOOT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
DBD=BE1PARTS:BE2PARTS
DBD=BE1ORDER:BE2ORDER
DBD=BE1LORDR:BE2LORDR
/*
```

Figure 62. Example of creating a DBD Compare report—Comparing two DBDs that have different names
Example: Comparing a DBD with a DBD-type ACB

The following figure shows example JCL for running the DBD/PSB/ACB Compare utility to compare several DBDs in DBD libraries IMSVS.DBDLIB with their corresponding DBD-type ACBs in ACB library IMSVS.ACBLIB.

```
//stepname EXEC PGM=FABLCOMP
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
  DBD=BE1PARTS,ACB
  DBD=BE2LORDR,ACB
  DBD=BE2ORDER,ACB
  DBD=BE2PARTS,ACB
/*
```

Figure 63. Example of creating a DBD Compare report—Comparing a DBD with DBD-type ACB

Example: Comparing two PSBs

The following figure shows example JCL for running the DBD/PSB/ACB Compare utility to compare several PSBs in PSB libraries IMSVS.PSBLIB and IMSVS.TEST.PSBLIB.

```
//stepname EXEC PGM=FABLCOMP
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//PSBLIB DD DSN=IMSVS.TEST.PSBLIB,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSOUT DD SYSOUT=A
//SYSIN DD *
  PSB=PE1CPINV
  PSB=PE1CPPUR
  PSB=PE1PPINV
  PSB=PE2CORDR
/*
```

Figure 64. Example of creating a PSB Compare report—Comparing two PSBs
Example: Comparing a PSB with a PSB-type ACB

The following figure shows example JCL for running the DBD/PSB/ACB Compare utility to compare several PSBs in PSB library IMSVS.PSBLIB with their corresponding PSB-type ACBs in ACB library IMSVS.ACBLIB.

```
//stepname EXEC PGM=FABLCOMP
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//ACBLIB2 DD DSN=IMSVS.ACBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSIN DD *
  PSB=PE1CPINV,ACB
  PSB=PE1CPPUR,ACB
  PSB=PE1PPINV,ACB
  PSB=PE2CORDR,ACB
/*
```

Figure 65. Example of creating a PSB Compare report—Comparing a PSB with PSB-type ACB

Example: Comparing two ACBs

The following figure shows example JCL for running the DBD/PSB/ACB Compare utility to compare several ACBs in ACB libraries IMSVS.ACBLIB and IMSVS.TEST.ACBLIB.

```
//stepname EXEC PGM=FABLCOMP
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//ACBLIB2 DD DSN=IMSVS.TEST.ACBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSIN DD *
  ACB=PE1CPINV
  ACB=PE1CPPUR
  ACB=PE1PPINV
  ACB=PE2CORDR
/*
```

Figure 66. Example of creating an ACB Compare report—Comparing two ACBs
Example: Comparing ACBs with DBDs and PSBs

The following figure shows example JCL for running the DBD/PSB/ACB Compare utility to compare several ACBs in ACB library IMSVS.ACBLIB with their corresponding DBDs in DBD library IMSVS.DBDLIB and PSBs in PSB library IMSVS.PSBLIB.

```plaintext
//stepname EXEC PGM=FABLCOMP
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//DBDLIB2 DD DSN=IMSVS.DBDLIB,DISP=SHR
//PSBLIB2 DD DSN=IMSVS.PSBLIB,DISP=SHR
//SYSCOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSSIN DD *
   ACB=BE1PARTS,BOTH
   ACB=BE2LORDR,BOTH
   ACB=PE1PPINV,BOTH
   ACB=PE2CORDR,BOTH
/*

Figure 67. Example of creating an ACB Compare report—Comparing ACBs with DBDs and PSBs

Example: Comparing DBDs, PSBs, and ACBs

The following figure shows example JCL for running the DBD/PSB/ACB Compare utility to compare several DBDs, PSBs, and ACBs.

```plaintext
//stepname EXEC PGM=FABLCOMP
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//PSBLIB2 DD DSN=IMSVS.PSBLIB,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//ACBLIB2 DD DSN=IMSVS.TEST.ACBLIB,DISP=SHR
//SYSCOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSSIN DD *
   DBD=BE1PARTS
   PSB=PE1CPPUR
   ACB=PE1PPINV
   DBD=BE2LORDR
   PSB=PE1CPPUR
   ACB=PE1CPINV
   DBD=BE2PARTS
   PSB=PE1PPINV
   PSB=PE2CORDR
   ACB=PE1PPINV
   ACB=PE2CORDR
/*

Figure 68. Example of creating a DBD, PSB, and ACB Compare report
Output from the DBD/PSB/ACB Compare utility

Output from the DBD/PSB/ACB Compare utility consists of the SYSOUT data set and the SYSPRINT data set.

SYSOUT data set

The SYSOUT data set contains a message for each input dbdname, psbname, and acbname. Each message states that the specified DBD, PSB, or ACB has been selected or not found in the DBD, PSB, or ACB library. It also contains all error messages. The summary of comparison, for each DBD, PSB, or ACB control statement, is also generated in the SYSOUT data set.

The following figure shows messages that are generated in the SYSOUT data set.

SYSPRINT data set

The SYSPRINT data set contains reports that are classified as DBD, PSB, and ACB Compare, and each group is sorted alphabetically by member name.

The SYSPRINT data set must contain records of 133 bytes or a multiple of 133.

DBD/PSB/ACB Compare generates the following types of reports:

Block-level compare reports

DBD/PSB/ACB Compare compares two DBDs, PSBs, or ACBs, and generates DBD, PSB, or ACB Compare reports that contain comparison information about two DBDs, PSBs, or ACBs.

Source-level compare reports

DBD/PSB/ACB Compare compares two control blocks (DBDs, PSBs, or ACBs) of the same type or different types at their source levels, and generates DBD, PSB, or ACB Compare reports that contain comparison information about the two control blocks.
Control Statement report

If you specify the CTLSTMT parameter for the REPORT statement, the utility generates the Control Statement report in the SYSPRINT data set. This report shows the echo of the SYSIN control statements and the selected runtime options.

The following figure shows an example of the Control Statement report.

---

Block-level compare reports

The block-level compare reports contain comparison results for two DBDs, PSBs, or ACBs.

Subsections:

- "Report field description"
- "Sample report: Block-level compare report for DBDs" on page 233
- "Sample report: Block-level compare report for PSBs" on page 234
- "Sample report: Block-level compare reports for ACBs" on page 235

Report field description

The blocks in DBD/PSB/ACB in LIBRARY 1 are taken as the basis for the comparisons. If an entry of a table such as segment table (SEGTAB) and field table (FLDTAB) are found in either LIBRARY 1 or LIBRARY 2, all information contained in the entry is reported. If the entry is found only in LIBRARY 2, an asterisk (*) appears to the right of the contents of LIBRARY 2.

The headings of DBD, PSB, and ACB Compare reports contain the following items in common:

**TYPE** Function type (DBD, PSB, or ACB) specified with the control statement.

**NAME** Name of the member or members that were compared. When the compared members have different names, the second member name follows the first member name, separated by a colon.

**LIBRARY 1** Data set name and volume serial number of the library that contains the member. Corresponds to DBDLIB, PSBLIB, or ACBLIB DD statements.
LIBRARY 2

Data set name and volume serial number of the library that contains the member. Corresponds to DBDLIB2, PSBLIB2, or ACBLIB2 DD statements.

SECTIONS WHICH ARE DIFFERENT

List of sections in which the differences were found.

DBD/PSB/ACB Compare compares two DBDs, PSBs, or ACBs that have the same name but are in different libraries. It also compares DBDs, PSBs, or ACBs that have different names and that are in the same library or in different libraries. In each pair, it compares the sections summarized in the following tables.

Table 16. Sections in DBD Compare reports

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIR</td>
<td>Information about the construction of the DBD control block</td>
</tr>
<tr>
<td>PRFX DB (dbname)</td>
<td>Database information</td>
</tr>
<tr>
<td>PRFX DSG (ddname)</td>
<td>Data set information</td>
</tr>
<tr>
<td>PRFX DSG (dsg-num)</td>
<td>Data set information for HALDB</td>
</tr>
<tr>
<td>SEGTAB SEG (segname)</td>
<td>Segment information</td>
</tr>
<tr>
<td>CMPRTN SEG (segname) CMP (exitname)</td>
<td>Compression exit information</td>
</tr>
<tr>
<td>FLDTAB SEG (segname) FLD (fldname)</td>
<td>Field information</td>
</tr>
<tr>
<td>LCHLD LCH (lchname)</td>
<td>Logical child information</td>
</tr>
<tr>
<td>EXTDBD EXT (extdbname)</td>
<td>External DB information that is referred to by the DBD</td>
</tr>
<tr>
<td>INDXTB</td>
<td>Index information</td>
</tr>
<tr>
<td>SSPTAB</td>
<td>Subset pointer information about DEDB DBD</td>
</tr>
<tr>
<td>SORTAB SEG (segname)</td>
<td>Source segment information</td>
</tr>
<tr>
<td>RDMRTN</td>
<td>HDAM randomizing routine information about HDAM DBD (see note)</td>
</tr>
<tr>
<td>DBDXTB</td>
<td>DBD extensional information</td>
</tr>
<tr>
<td>SEGUXTB SEG (segname)</td>
<td>Segment exit table information</td>
</tr>
<tr>
<td>EXITTB SEG (segname) EXT (exitname)</td>
<td>Exit name array information</td>
</tr>
<tr>
<td>INDXMP PSL (exitname)</td>
<td>FPSI Partition Selection exit information</td>
</tr>
</tbody>
</table>

Chapter 6. DBD/PSB/ACB Compare utility  229
### Table 16. Sections in DBD Compare reports (continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DXVECT</td>
<td>Metadata information for DBD</td>
</tr>
<tr>
<td>DXDENT</td>
<td></td>
</tr>
<tr>
<td>DXDRET</td>
<td></td>
</tr>
<tr>
<td>DXSENT SEG (segname)</td>
<td></td>
</tr>
<tr>
<td>DXSEXT SEG (segname)</td>
<td></td>
</tr>
<tr>
<td>DXSRET</td>
<td></td>
</tr>
<tr>
<td>DXFDTT</td>
<td></td>
</tr>
<tr>
<td>DXFDOT</td>
<td></td>
</tr>
<tr>
<td>DXFXTT</td>
<td></td>
</tr>
<tr>
<td>DXFRDT</td>
<td></td>
</tr>
<tr>
<td>DXFRET</td>
<td></td>
</tr>
<tr>
<td>DXFPAT</td>
<td></td>
</tr>
<tr>
<td>DXFCIT</td>
<td></td>
</tr>
<tr>
<td>DXFCAT</td>
<td></td>
</tr>
<tr>
<td>DXFCRT</td>
<td></td>
</tr>
<tr>
<td>DXFMMDT</td>
<td></td>
</tr>
<tr>
<td>DXFMNT</td>
<td></td>
</tr>
<tr>
<td>DXFSAT</td>
<td></td>
</tr>
<tr>
<td>DXMENT</td>
<td></td>
</tr>
<tr>
<td>DXMPAT</td>
<td></td>
</tr>
<tr>
<td>DXMOVT</td>
<td></td>
</tr>
<tr>
<td>DXMFFT</td>
<td></td>
</tr>
<tr>
<td>DXMFRT</td>
<td></td>
</tr>
<tr>
<td>DXFSAT</td>
<td></td>
</tr>
<tr>
<td>DXMPAT</td>
<td></td>
</tr>
<tr>
<td>DXMURT</td>
<td></td>
</tr>
<tr>
<td>DXMREY</td>
<td></td>
</tr>
<tr>
<td>DXLRET</td>
<td></td>
</tr>
<tr>
<td>DXMRET</td>
<td></td>
</tr>
<tr>
<td>DXMRT</td>
<td></td>
</tr>
<tr>
<td>DXMPOT</td>
<td></td>
</tr>
<tr>
<td>DXMPOE</td>
<td></td>
</tr>
<tr>
<td>DXTRET</td>
<td></td>
</tr>
<tr>
<td>DXARET</td>
<td></td>
</tr>
<tr>
<td>DXFEXT FLD (fldname)</td>
<td></td>
</tr>
<tr>
<td>DXDRET</td>
<td></td>
</tr>
<tr>
<td>DBDGEN</td>
<td>IMS release level information. If the DBD was generated by IMS Version 3 or higher, the level is shown.</td>
</tr>
<tr>
<td>CKTBL1</td>
<td>/CK search field information about DEDB DBD</td>
</tr>
<tr>
<td>CKTBL2</td>
<td>/CK subsequence field information about DEDB DBD</td>
</tr>
</tbody>
</table>

**Note:** The differences in this section are checked only if the RDMRTN section is customized. For details about customization, see the topic "HDAM and PHDAM randomizing routines (DFSHDC40)” in the *IMS Exit Routines*.

### Table 17. Sections in PSB Compare reports

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRFX</td>
<td>PSB attributes information</td>
</tr>
<tr>
<td>TP PCB (ltrname)</td>
<td>TP PCB information</td>
</tr>
<tr>
<td>DB PCB (dbname)</td>
<td>DB PCB information</td>
</tr>
<tr>
<td>GS PCB (dbname)</td>
<td>GSAM DB PCB information</td>
</tr>
<tr>
<td>SENSEG PCB (dbname) SEG (segname)</td>
<td>Sensitive segment information about DB PCB</td>
</tr>
<tr>
<td>PSSPTB PCB (dbname) SEG (segname)</td>
<td>Subset pointer information about DB PCB</td>
</tr>
</tbody>
</table>
### Table 17. Sections in PSB Compare reports (continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENFLD PCB ((dbname)) SEG ((segname)) FLD ((fldname))</td>
<td>Sensitive field information about DB PCB</td>
</tr>
<tr>
<td>REFTBL DB ((dbname))</td>
<td>DB information referred to by DB PCB or GSAM PCB</td>
</tr>
<tr>
<td>PCBNAME NAM ((pcbname))</td>
<td>PCB name information specified in the PCB statement</td>
</tr>
<tr>
<td>PXVEXT</td>
<td>Metadata information for PSB</td>
</tr>
<tr>
<td>PXPCRT</td>
<td></td>
</tr>
<tr>
<td>PXFCXT</td>
<td></td>
</tr>
<tr>
<td>PXSSRT</td>
<td></td>
</tr>
<tr>
<td>PXSFRT</td>
<td></td>
</tr>
<tr>
<td>PXPSRT</td>
<td></td>
</tr>
<tr>
<td>PSBGEN</td>
<td>IMS release level information. If the PSB was generated by IMS Version 3 or higher, the level is shown.</td>
</tr>
</tbody>
</table>

### Table 18. Sections in ACB Compare reports

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRFX</td>
<td>ACB attributes and database information</td>
</tr>
<tr>
<td>TPPCB PCB ((ltrmname))</td>
<td>TP PCB information</td>
</tr>
<tr>
<td>PCBNAME PCB ((ltrmname)) NAM ((pcbname))</td>
<td>PCB name information about TP PCB</td>
</tr>
<tr>
<td>DBPCB PCB ((dbname))</td>
<td>DB PCB information</td>
</tr>
<tr>
<td>DBPCBX PCB ((dbname))</td>
<td>DB PCB extensional information</td>
</tr>
<tr>
<td>VERID PCB ((dbname))</td>
<td>Version ID information about the DBD that is referred to by DB PCB</td>
</tr>
<tr>
<td>DMIBXTB PCB ((dbname))</td>
<td>DMB extensional information about DEDB DMB referred to by DB PCB</td>
</tr>
<tr>
<td>SEGXTB PCB ((dbname)) SEG ((segnam)</td>
<td>Segment exit table information about DEDB DMB referred to by DB PCB</td>
</tr>
<tr>
<td>EXITTB PCB ((dbname)) SEG ((segnam) EXT((exitname))</td>
<td>Exit name array information about DEDB DMB referred to by DB PCB</td>
</tr>
<tr>
<td>JCB PCB ((pcbname))</td>
<td>JOB control block information</td>
</tr>
<tr>
<td>SDB SEG ((segnam)</td>
<td>Sensitive segment information about DB PCB</td>
</tr>
<tr>
<td>FSBLST</td>
<td>Information related to sensitive field of DB PCB</td>
</tr>
<tr>
<td>FSB FLD ((fldname))</td>
<td>Sensitive field information about DB PCB</td>
</tr>
<tr>
<td>SPCB</td>
<td>DB PCB information for DEDB DBD</td>
</tr>
<tr>
<td>SMLT SEG ((segnam)</td>
<td>Sensitive segment information about DB PCB for DEDB DBD</td>
</tr>
<tr>
<td>RDMIBT</td>
<td>HDAM randomizing routine information about HDAM DBD</td>
</tr>
<tr>
<td>AMPBPX</td>
<td>Prefix information about the Access Method Prefix Block</td>
</tr>
<tr>
<td>AMPB</td>
<td>Access Method Prefix Block information</td>
</tr>
<tr>
<td>PSDB</td>
<td>Physical Segment Descriptor Block information</td>
</tr>
<tr>
<td>SECOND SEG ((segnam)</td>
<td>Secondary list information</td>
</tr>
<tr>
<td>FDB FLD ((fldname))</td>
<td>Field Description Block information</td>
</tr>
<tr>
<td>DMIBXTB</td>
<td>DMB extensional information</td>
</tr>
<tr>
<td>SEGXTB SEG ((segnam)</td>
<td>Segment exit table information</td>
</tr>
<tr>
<td>EXITTB SEG ((segnam) EXT ((exitname))</td>
<td>Exit name array information</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CPAC CMP (exitname)</td>
<td>Compression exit information</td>
</tr>
<tr>
<td>DMCB DBD (dbdname)</td>
<td>DEDB Master Control Block information about DEDB DBD</td>
</tr>
<tr>
<td>SDT SEG (segname)</td>
<td>Segment information about DEDB DBD</td>
</tr>
<tr>
<td>FDT SEG (segname) FLD (fldname)</td>
<td>Field information about DEDB DBD</td>
</tr>
<tr>
<td>MRMB DBD (dbdname)</td>
<td>DEDB Randomizing Module Block information about DEDB DBD</td>
</tr>
<tr>
<td>DMAC DBD (dbdname) DSG (areaname)</td>
<td>DEDB AREA Control List information about DEDB DBD</td>
</tr>
<tr>
<td>BHDR DBD (dbdname) FLD (fldname)</td>
<td>Header information about MSDB DBD</td>
</tr>
<tr>
<td>CRTE</td>
<td>DEDB secondary index cross reference table entries information</td>
</tr>
<tr>
<td>CRTEXD</td>
<td>INDEX record partitioning information</td>
</tr>
</tbody>
</table>

ACB Compare reports also contain metadata information for DBDs and PSBs. For those sections, see Table 16 on page 229 and Table 17 on page 230.

**DIFFERENCE DESCRIPTION**

Description of the field in which difference was found.

When differences are found between two control blocks after running DBD/PSB/ACB Compare, you can run DBD/PSB/ACB Mapper, or DBD/PSB/ACB Reversal to obtain more information about the control blocks.

**Note:** Any comparison of two variable-length fields in the IMS control blocks is based on the field length of the LIBRARY 1 block. If any difference is found, DBD/PSB/ACB Compare reports it in full length, using one or more lines in the LIBRARY 1 and LIBRARY 2 fields.
Sample report: Block-level compare report for DBDs

The following figure shows an example of the DBD Compare report.

<table>
<thead>
<tr>
<th>RECORD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st and 2nd records</td>
<td>In the prefix section, different access methods are specified: HIDAM VSAM in LIBRARY 1, and HIDAM OSAM in LIBRARY 2.</td>
</tr>
<tr>
<td>3rd record</td>
<td>In the prefix section, different DD names are specified: DSFACH00 in LIBRARY 1, and DSFACH00 in LIBRARY 2.</td>
</tr>
<tr>
<td>10th record</td>
<td>In segment SSFACP00 in the SEGTAB section, different segment lengths are specified: 35 in LIBRARY 1, and 30 in LIBRARY 2.</td>
</tr>
<tr>
<td>11th record</td>
<td>In the FLDTAB section, different field names are specified for the ETEANA00 field of segment SSFACP00: ETEANA00 in LIBRARY 1, and ETEANAME in LIBRARY 2.</td>
</tr>
</tbody>
</table>

The records in the figure have the following meanings:

Figure 71. DBD Compare report

<table>
<thead>
<tr>
<th>LIBRARY1</th>
<th>LIBRARY2</th>
<th>SECTIONS WHICH ARE DIFFERENT</th>
<th>DIFFERENCE DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>YES</td>
<td>PRFX DB (DSFACHON)</td>
<td>HIDAM OSAM</td>
</tr>
<tr>
<td>YES</td>
<td>NO</td>
<td>PRFX DB (DSFACHON)</td>
<td>HIDAM VSAM</td>
</tr>
<tr>
<td>DSFACH00</td>
<td>DSFACH00</td>
<td>PRFX DSG(DSFACH00)</td>
<td>INPUT DD NAME/MSDB SEQUENCED FIELD NAME</td>
</tr>
<tr>
<td>3380</td>
<td>3350</td>
<td>PRFX DSG(DSFACH00)</td>
<td>DEVICE TYPE OR RESERVED FIELD</td>
</tr>
<tr>
<td>57</td>
<td>52</td>
<td>PRFX DSG(DSFACH00)</td>
<td>LONGEST SEGMENT LENGTH</td>
</tr>
<tr>
<td>2041</td>
<td>0</td>
<td>PRFX DSG(DSFACH00)</td>
<td>LOGICAL RECORD LENGTH</td>
</tr>
<tr>
<td>2048</td>
<td>1690</td>
<td>PRFX DSG(DSFACH00)</td>
<td>BLOCK/CI SIZE</td>
</tr>
<tr>
<td>35</td>
<td>30</td>
<td>SEGTAB SEG(SSFACP00)</td>
<td>OVERFLOW/OUTPUT BLOCK/CI SIZE</td>
</tr>
<tr>
<td>ETEANA00</td>
<td>ETEANAME</td>
<td>FLDTAB SEG(SSFACP00)</td>
<td>DATA LEN + SEG LEN FOR FIXED LEN SEGS</td>
</tr>
<tr>
<td>NO</td>
<td>YES</td>
<td>INDXTB</td>
<td>FIELD/RESEED NAME</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>INDEX POINTER IS SYMBOLIC</td>
</tr>
</tbody>
</table>

Chapter 6. DBD/PSB/ACB Compare utility 233
Sample report: Block-level compare report for PSBs

The following figure shows an example of the PSB Compare report.

<table>
<thead>
<tr>
<th>LIBRARY1</th>
<th>LIBRARY2</th>
<th>SECTIONS WHICH ARE DIFFERENT</th>
<th>DIFFERENCE DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>YES</td>
<td>PRFX</td>
<td>APPLICATION PROGRAM IS ASSEMBLER, COBOL OR NOT SPECIFIED</td>
</tr>
<tr>
<td>YES</td>
<td>NO</td>
<td>PRFX</td>
<td>APPLICATION PROGRAM IS PL/I</td>
</tr>
<tr>
<td>GID A</td>
<td></td>
<td>DBPCB PCB(DSFACHON)</td>
<td>PROCESSING OPTIONS</td>
</tr>
<tr>
<td>GID A</td>
<td></td>
<td>SENSEG PCB(DSFACHON)</td>
<td>PCB(DSFACHON)</td>
</tr>
<tr>
<td>GID A</td>
<td>140</td>
<td>SENSEG PCB(DSSTUIVN)</td>
<td>PCB(DSSTUIVN)</td>
</tr>
<tr>
<td>A</td>
<td>140</td>
<td>SENSEG PCB(DSSTUIVN)</td>
<td>PCB(DSSTUIVN)</td>
</tr>
</tbody>
</table>

Figure 72. PSB Compare report

The records in the figure have the following meaning:

1st and 2nd records
   In the prefix section, different application program languages are specified: PL/I in LIBRARY 1, and Assembler or COBOL in LIBRARY 2.

3rd record
   In the DBPCB section, different processing options are specified in the DSFACHON PCB: GID in LIBRARY 1, and A in LIBRARY 2.

4th through 6th records
   In the sensitive segments of DSFACHON in the SENSEG section, different processing options are specified: GID in LIBRARY 1, and A in LIBRARY 2.
Sample report: Block-level compare reports for ACBs

The following figure shows an example of the ACB Compare report.

The records in the figure have the following meaning:

2nd and 3rd records
In the prefix section, different access methods are specified: HIDAM VSAM in LIBRARY 1, and HD INDEXED in LIBRARY 2.

4th record
In the AMPBPX section, the lengths of largest segment in the data set are different: 57 in LIBRARY 1, and 52 in LIBRARY 2.

9th record
In the PSDB section, the data lengths of the segment are different: 35 in LIBRARY 1, and 30 in LIBRARY 2.

The following figure shows another example for the ACB Compare report.
The records in the figure have the following meaning:

**1st and 2nd records**
In the prefix section, different application program languages are specified: PL/I in LIBRARY 1, and Assembler or COBOL in LIBRARY 2.

**3rd record**
In the DBPCB section, different processing options are specified in DSSCHHVN PCB: GID in LIBRARY 1, and A in LIBRARY 2.

**4th record**
In the DBPCB section, different numbers of the sensitive segments in DSSCHHVN PCB are given: three in LIBRARY 1, and four in LIBRARY 2.

### Source-level compare reports

The source-level compare reports contain comparison results for comparing two control blocks at the source level.

**Subsections:**
- [“Report field description”](#)
- [“Sample report: Source-level compare report for DBDs”](#)
- [“Sample report: Source-level compare report for PSBs”](#)
- [“Sample report: Source-level compare reports for ACBs”](#)

#### Report field description

The sources of DBD, PSB, or ACB in LIBRARY 1 are taken as the basis for the comparisons if a source-level compare report is generated.
The headings of DBD, PSB, and ACB Compare reports contain the following items in common:

**TYPE**   Function type (DBD, PSB, ACB, DBDACB, or PSBACB).

**NAME**   Name of the member or members that were compared. When the compared members have different names, the second member name follows the first member name, separated by a colon.

**LIBRARY 1**   Data set name and volume serial number of the library that contains the member. Corresponds to DBDLIB, PSBLIB, or ACBLIB DD statements.

**LIBRARY 2**   Data set name and volume serial number of the library that contains the member. Corresponds to DBDLIB2, PSBLIB2, or ACBLIB2 DD statements.

**NUMBER OF DIFFERENT STATEMENTS**   This part contains the summary information about statements which were inserted, deleted, or changed.

**INSERTED**   The number of statements which were found only in DBD, PSB, or ACB in LIBRARY 2.

**DELETED**   The number of statements which were found only in DBD, PSB, or ACB in LIBRARY 1.

**CHANGED**   The number of statements which were found in both DBD, PSB, or ACB in LIBRARY 1 and DBD, PSB, or ACB in LIBRARY 2, but were detected to be different.

**DBDGEN/PSBGEN/ACBGEN**   The date and time when the DBD/PSB/ACB was generated.

**IMSREL**   The IMS version and release that generated the DBD/PSB/ACB.

**CHK**   The following characters are shown if any difference is found in DBDs/PSBs/ACBs between LIBRARY 1 and LIBRARY 2:

- **I**   When a statement is inserted into DBD/PSB/ACB in LIBRARY 2.
- **D**   When a statement is deleted from DBD/PSB/ACB in LIBRARY 1.
- **C**   When a statement in DBD/PSB/ACB in LIBRARY 1 is different from that in LIBRARY 2. An asterisk (*) is shown on the row of each data which is determined to be different.

**LIBRARY 1 SOURCE LINES**   The IMS DBDGEN or PSBGEN utility control statements which were decoded from DBD/PSB/ACB in LIBRARY 1.

**LIBRARY 2 SOURCE LINES**   The IMS DBDGEN or PSBGEN utility control statements which were decoded from DBD/PSB/ACB in LIBRARY 2.
Sample report: Source-level compare report for DBDs

The following figure is an example of a DBD Compare report.

---

**Sample report: Source-level compare report for DBDs**

The following figure is an example of a DBD Compare report.

---

**IMS Library Integrity Utilities - DBD/PSB/ACB Compare**

<table>
<thead>
<tr>
<th>TYPE</th>
<th>DBD</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>DBD0003A</td>
</tr>
</tbody>
</table>

**LIBRARY 1**: VOLUME=IMSVS DSNAME=IMSVS.DBDLIB

**LIBRARY 2**: VOLUME=IMSVS DSNAME=IMSVS.TEST.DBDLIB

**NUMBER OF DIFFERENT STATEMENTS**

<table>
<thead>
<tr>
<th>INSERTED</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELETED</td>
<td>0</td>
</tr>
<tr>
<td>CHANGED</td>
<td>3</td>
</tr>
</tbody>
</table>

**DBGEN**: 08/04/2015 17:41

**IMSREL**: 1310

**CHK LIBRARY 1 SOURCE LINES**

---

<table>
<thead>
<tr>
<th>C - DBD</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME=DD0003A, ACCESS=(HDAM,OSAM), RNAME=(RMM,2,500,800), PASSWD=NO, VERSION= 08/04/15 17.41</td>
</tr>
</tbody>
</table>

**Library 2 Source Lines**

---

<table>
<thead>
<tr>
<th>C - DBD</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME=DD0003A, ACCESS=(HDAM,VSAM), RNAME=(RMM,2,500,800), PASSWD=NO, VERSION= 08/04/15 17.41</td>
</tr>
</tbody>
</table>

**C - DSG1 DATASET**

| DDI=DD0003A, SIZE=(1690), SCAN=3, FRSPC=(2,3) |

**Library 2 Source Lines**

---

<table>
<thead>
<tr>
<th>C - DSG1 DATASET</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDI=DD0003A, SIZE=(2048), SCAN=3, FRSPC=(2,3)</td>
</tr>
</tbody>
</table>

**C - SEGM**

| NAME=DDSEG1, PARENT=0, BYTES=100, RULES=(LLL,LAST), PTR=(TWIN,...) |

**Library 2 Source Lines**

---

<table>
<thead>
<tr>
<th>I - SEGM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME=DDSEG2, PARENT=((DDSEG1,)), BYTES=100, RULES=(LLL,LAST), PTR=(TWIN,...)</td>
</tr>
</tbody>
</table>

**Library 2 Source Lines**

---

<table>
<thead>
<tr>
<th>I - SEGM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME=DDSEG2, PARENT=((DDSEG1,)), BYTES=100, RULES=(LLL,LAST), PTR=(TWIN,...)</td>
</tr>
</tbody>
</table>

**FIELD**

| NAME=(D03FLD1A,SEQ,U), START=1, BYTES=10, TYPE=C |

**Library 2 Source Lines**

---

<table>
<thead>
<tr>
<th>I - FIELD</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME=(D03FLD1B), START=11, BYTES=10, TYPE=X</td>
</tr>
</tbody>
</table>

**Library 2 Source Lines**

---

<table>
<thead>
<tr>
<th>I - FIELD</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME=(D03FLD2A,SEQ,U), START=1, BYTES=10, TYPE=C</td>
</tr>
</tbody>
</table>

**Library 2 Source Lines**

---

<table>
<thead>
<tr>
<th>I - FIELD</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME=(D03FLD2A,SEQ,U), START=1, BYTES=10, TYPE=C</td>
</tr>
</tbody>
</table>

**DBDGEN**

<table>
<thead>
<tr>
<th>FINISH</th>
</tr>
</thead>
<tbody>
<tr>
<td>END</td>
</tr>
</tbody>
</table>

**Figure 75. DBD source-level compare report**

---

IMS Library Integrity Utilities User’s Guide
Sample report: Source-level compare report for PSBs

The following figure is an example of a PSB Compare report.

Figure 76. PSB source-level compare report
Sample report: Source-level compare reports for ACBs

The following figure is an example of an ACB Compare report.

---

Figure 77. ACB source-level compare report (Sample 1)

The following figure is another example of an ACB Compare report.
Figure 78. ACB source-level compare report (Sample 2)
Chapter 7. DBD/PSB/ACB Mapper utility

The DBD/PSB/ACB Mapper utility produces printed maps (pictures of the segment hierarchy) from DBDs, PSBs, and ACBs. The utility also produces detailed reports that describe DBDs, PSBs, and ACBs.

Topics:
- “DBD/PSB/ACB Mapper utility overview” on page 244
- “Restrictions for the DBD/PSB/ACB Mapper utility” on page 246
- “Printing hierarchical structure of databases” on page 247
- “JCL requirements for the DBD/PSB/ACB Mapper utility” on page 248
- “Control statements for the DBD/PSB/ACB Mapper utility” on page 250
- “JCL examples for the DBD/PSB/ACB Mapper utility” on page 253
- “Output from the DBD/PSB/ACB Mapper utility” on page 259
The DBD/PSB/ACB Mapper utility, a productivity aid, can produce and print a pictorial layout, called a map, that graphically represents the structure and characteristics of a physical and logical IMS database. The DBD/PSB/ACB Mapper utility can also print a detailed report describing the characteristics of each database description (DBD).

Subsections:
- “Function overview”
- “Program structure and job step”
- “Data flow” on page 245

Function overview

The utility provides the following functions:

DBD Map function
The DBD Map function reads one or more DBDs from DBD libraries and produces maps and reports for the DBDs.

PSB Map function
The PSB Map function reads one or more PSBs from PSB libraries and produces maps and reports for the PSBs.

ACB Map function
The ACB Map function reads one or more ACBs from ACB libraries and produces maps and reports for the ACBs.

The complete visual representation can be used as a recording medium to retain the historical and current status of the IMS databases. The maps can also be used as a reference in comparing and evaluating the database requirements of current and proposed applications.

DBD/PSB/ACB Mapper supports all the IMS access methods except the mapping support of GSAM.

DBD/PSB/ACB Mapper can be set to run each time an IMS database control block is changed. This procedure ensures that a current picture and description of the database are produced each time the structure is changed.

Related reading: The following topics provide JCL examples for creating a DBD or a PSB and generating a map:
- “Example: Creating a DBD and generating a DBD map” on page 257
- “Example: Creating a PSB and generating a PSB map” on page 258

Program structure and job step

DBD/PSB/ACB Mapper consists of one program, FABMMMAIN. This program controls load modules FABMDMAP, FABMFMAP, and FABMAMAP. The FABMMMAIN program builds and prints a report of IMS control blocks selected from a DBD, PSB, or ACB library, and prints maps of the physical and logical IMS database definitions.

FABMMMAIN uses a simple input format that is specified in the SYSIN data set. The names of the DBDs, PSBs, or ACBs selected for mapping or reporting are specified
in the SYSIN data set control statements. The program attempts to find the DBD, PSB, or ACB in the data sets defined by the DD statement. If it cannot find them, it posts a notice on the activity log (SYSOUT DD statement). Processing then continues until all of the input control statements are processed.

**Data flow**

The general data flow for DBD/PSB/ACB Mapper (FABMMAIN) is shown in the following figure. Input consists of the SYSIN data set and the DBDLIB, PSBLIB, and ACBLIB data sets. Output consists of the DBD, PSB, and ACB reports, DBD, PSB, and ACB maps, and the activity log.

---

![Figure 79. Data flow for DBD/PSB/ACB Mapper](image-url)
Restrictions for the DBD/PSB/ACB Mapper utility

Certain restrictions apply when you use the DBD/PSB/ACB Mapper utility.

DBD/PSB/ACB Mapper can produce a map and a report for the DBD of any IMS database organization, except GSAM. Because GSAM has no segment, only a report can be prepared.

If the reports of IMS Library Integrity Utilities are printed on a 3800 printer, text fonts GT15 and ST15 used at 12 lines per inch (LPI) do not work. The reason is that the underscore (_) character, which is used to draw the horizontal lines for the DBD hierarchy, are not printed. Fonts GSC and GFC can be used. The GT15 and ST15 fonts can be used at 6 and 8 LPI.

When the parameter values added by certain versions and releases of IMS are the default values, the fields for those parameters are not printed in the report for DBD or PSB.
Printing hierarchical structure of databases

To generate reports and print hierarchical structure of databases by using the DBD/PSB/ACB Mapper utility, you must prepare the JCL for the DBD/PSB/ACB Mapper utility and submit the job.

About this task

Sample JCL for the DBD/PSB/ACB Mapper utility is in the SHPSJCL0 library, member FABLIVP1. You can modify this sample JCL and then use it to run the utility.

Procedure

1. In the DBD/PSB/ACB Mapper JCL, code the EXEC statement and DD statements.
   See “JCL requirements for the DBD/PSB/ACB Mapper utility” on page 248.
2. In the SYSIN data set, code the control statements for the DBD/PSB/ACB Mapper utility.
   See “Control statements for the DBD/PSB/ACB Mapper utility” on page 250.
3. Submit the job.
4. Check the output data sets that are generated.
   See “Output from the DBD/PSB/ACB Mapper utility” on page 259.
**JCL requirements for the DBD/PSB/ACB Mapper utility**

When you code the JCL to run the DBD/PSB/ACB Mapper utility, include the EXEC statement and appropriate DD statements.

**Subsections:**
- “JCL example”
- “EXEC statement”
- “DD statements”

**JCL example**

An example of the JCL that is required for DBD/PSB/ACB Mapper is shown in the following figure.

```plaintext
//stepname EXEC PGM=FABMMAIN,REGION=512K
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *  
  (control statements)
/*
```

*Figure 80. Example of DBD/PSB/ACB Mapper JCL (FABMMAIN JCL)*

**EXEC statement**

This statement must have the following format:

```plaintext
//stepname EXEC PGM=FABMMAIN
```

**Note:** FABMMAIN does not allow EXEC statement parameters.

**DD statements**

Code the following DD statements to identify the source of input and the placement of output information:

**STEPLIB DD or JOBLIB DD**

This DD statement is required. This input DD statement defines the IMS Library Integrity Utilities load module library.

**DBDLIB DD**

This DD statement is required for the DBD Map function and the PSB Map function. The DBDLIB DD input data set is the library that contains the DBDs for which reports or maps are created.

**PSBLIB DD**

This DD statement is required for the PSB Map function. The PSBLIB DD input data set is the library that contains the PSBs for which reports are created.
ACBLIB DD
This DD statement is required for the ACB Map function. The ACBLIB DD input data set is the library that contains the ACBs for which reports are created.

SYSOUT DD
This DD statement is required. The SYSOUT DD output data set contains all activity messages and error messages. The record format is fixed-blocked and the logical record length is 133. The block size, if coded, must be a multiple of 133.

SYSPRINT DD
This DD statement is required. The SYSPRINT DD output data set contains the reports, maps, or both that are created by DBD/PSB/ACB Mapper. The reports are arranged in the order of DBD, PSB, and ACB, and the members in each group are sorted alphabetically. The record format is fixed-blocked and the logical record length is 133. The block size, if coded, must be a multiple of 133.

SYSIN DD
This DD statement is required. The SYSIN DD input data set contains the control statements for the DBD/PSB/ACB Mapper program. The record format is fixed-blocked and the logical record length is 80. The block size, if coded, must be a multiple of 80.

Up to 9999 control statements can be specified using the SYSIN DD statement. If there are more than 9999 control statements, the excess control statements are ignored.

Related reading: For the format of the control statements, see “Control statements for the DBD/PSB/ACB Mapper utility” on page 250.
Control statements for the DBD/PSB/ACB Mapper utility

The input to the DBD/PSB/ACB Mapper utility consists of control statements in the SYSIN data set. These control statements contain keywords that indicate the functions and the names of the DBDs, PSBs, or ACBs for which reports and maps are created.

This data set usually resides in the input data stream. However, it can be defined as a sequential data set or a member of a partitioned data set. It must contain one 80-byte fixed-length record for each DBD, PSB, and ACB processed. The block size, if coded, must be a multiple of 80.

The order in which the reports and maps are written to the data set is DBD, PSB, PSB-type ACB, and DBD-type ACB. The members in each group are ordered alphabetically.

Subsections:
- “Control statement example”
- “Syntax rules”
- “Control statement keywords” on page 251
- “Quick reference for control statements and DD statements” on page 252

Control statement example

The SYSIN data set can be coded as shown in the following figure.

```
//SYSIN DD *
  DBD=member
  DBD=member,X
  PSB=member
  PSB=member,X
  ACB=member
  ACB=member,X
  ACBDBD=member
  ACBDBD=member,X
/*
```

Figure 81. Examples of control statements for DBD/PSB/ACB Mapper

Syntax rules

The control statements for DBD/PSB/ACB Mapper must adhere to the following syntax rules:
- Control statements can be coded anywhere between columns 2 - 80.
- In the control statement field, keyword, equal sign, member name, comma, and X must not be separated by blanks. Because a blank serves as the delimiter, only a comment can be written after a blank.
- A statement with an asterisk (*) in column 1 is treated as a comment.
- The control statements can be specified in any order. For example, in the following order:
Control statement keywords

The control statement formats are as follows:

DBD=member
This control statement shows the format used for obtaining both the DBD map and the DBD report.

DBD=member,X
This control statement is used for obtaining only a DBD map. The member is followed by a comma and a non-blank character.

PSB=member
This control statement is used for obtaining the PSB Summary report, PSB Maps, and PSB reports of all DBDs relating to this PSB.

PSB=member,X
This control statement is used for obtaining the PSB Summary report and the PSB maps of all associated DBDs. The member is followed by a comma and a non-blank character.

ACB=member
This control statement is used to obtain the ACB (PSB) Summary report, ACB (PSB) Maps, and ACB (PSB) reports of all associated ACBs.

ACB=member,X
This control statement is used to obtain the ACB (PSB) Summary report and the ACB (PSB) Maps of all associated DBDs. The member is followed by a comma and a non-blank character.

Note: The ACB library member member following the ACB= keyword must be a PSB-type. Otherwise, an error message is issued and the processing of this member is skipped.

ACBDBD=member
This control statement is used to obtain the ACB (DBD) Maps and the ACB (DBD) report.

ACBDBD=member,X
This control statement is used to obtain only ACB (DBD) Maps. The member is followed by a comma and a non-blank character.

Note: The ACB library member member following the ACBDBD= keyword must be a DBD-type. Otherwise, an error message is issued and the processing of this member is skipped.

Note: You can specify a wildcard at any position in a character string. The asterisk (*) and the percent sign (%) are supported as wildcard characters. An asterisk (*) represents 0 - 8 characters, and a percent sign (%) represents a single character. If two or more asterisks (*) are specified sequentially, only the first asterisk is recognized.
Quick reference for control statements and DD statements

The following table lists the DBD/PSB/ACB Mapper functions, control statements, and DD statements.

Table 19. DBD/PSB/ACB Mapper functions, control statements, and DD statements

<table>
<thead>
<tr>
<th>Function</th>
<th>Control keyword</th>
<th>Required DD statements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SYSPRINT DD</td>
</tr>
<tr>
<td>DBD map</td>
<td>DBD=</td>
<td>Required</td>
</tr>
<tr>
<td>PSB map</td>
<td>PSB=</td>
<td>Required</td>
</tr>
<tr>
<td>ACB map</td>
<td>ACB=</td>
<td>Required</td>
</tr>
<tr>
<td></td>
<td>ACBDDB=</td>
<td>Required</td>
</tr>
</tbody>
</table>
JCL examples for the DBD/PSB/ACB Mapper utility

The following topics provide JCL examples for running the DBD/PSB/ACB Mapper utility to print maps of databases.

Example: Generating DBD maps

The following figure shows example JCL for running a job that generates maps of multiple DBDs.

If a dbdname is not followed by characters, both a map and report are produced for each DBD. If a dbdname is followed by ,X or a comma and any other character, only a map is produced for the DBD; the report for it is omitted.

```
//DBDMAP EXEC PGM=FABMAIN
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
    DBD=BE1PARTS,X
    DBD=BE2LORDR
    DBD=BE2LPART,X
    DBD=BE2ORDER
    DBD=BE2ORDRX,X
    DBD=BE2PARTS
    DBD=BE2PCUST,X
    DBD=B00INP01
    DBD=B00OUT01,X
/*
```

Figure 82. Example of generating DBD maps
Example: Generating PSB maps

The following figure shows example JCL for running a job that generates maps of multiple PSBs.

Each psbname produces a PSB Summary report, a map, and a report for a database PCB in the PSB. A psbname followed by ,X produces only a PSB Summary report for the PSB, and a map for each database PCB in that PSB. The X can be replaced by any other character.

```plaintext
//PSBMAP   EXEC PGM=FABMMAIN
//STEPLIB   DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBOLIB    DD DSN=IMSVS.DBDLIB,DISP=SHR
//PSBLIB    DD DSN=IMSVS.PSBLIB,DISP=SHR
//SYSOUT    DD SYSOUT=A
//SYSPRINT  DD SYSOUT=A
//SYSSN     DD *
PSB=PE1CPINV,X
PSB=PE1CPPUR
PSB=PE1PARTS,X
PSB=PE1PPINV
PSB=PE1PPPUR,X
PSB=PE2CORDR
PSB=PE2ORDER,X
PSB=PLIPROCS
/*
```

Figure 83. Example of generating PSB maps
Example: Generating ACB maps

The following figure shows example JCL for running a job that generates maps of multiple ACBs.

Each PSB-type ACB member specified by ACB=, if followed by no characters, produces an ACB (PSB) Summary report and both an ACB (PSB) map and an ACB (PSB) report for each database PCB within the PSB-type ACB. A PSB-type acbname followed by ,X produces only an ACB (PSB) Summary report and ACB (PSB) maps.

Each DBD-type ACB member specified by ACBDBD=, if followed by no characters, produces both an ACB (DBD) map and an ACB (DBD) report. A DBD-type acbname followed by ,X produces only an ACB (DBD) map.

The X can be replaced by any character.

//ACBMAP  EXEC  PGM=FABMAIN
//STEPLIB  DD  DSN=HPS.SHPSLMD0,DISP=SHR
//ACBLIB  DD  DSN=IMSVS.ACBLIB,DISP=SHR
//SYSOUT  DD  SYSOUT=A
//SYSPRINT  DD  SYSOUT=A
//SYSIN  DD  *
   ACB=PE1CPINV,X
   ACB=PE1CPPUR
   ACB=PE1PARTS,X
   ACB=PE1PPINV
   ACBDBD=DE1PPPUR,X
   ACBDBD=DE2CORDR
   ACBDBD=DE2ORDER,X
   ACBDBD=DLIPROCS
/*

Figure 84. Example of generating ACB maps
Example: Generating DBD, PSB, ACB maps

The following figure shows example JCL for running the function of DBD/PSB/ACB Mapper inner single job step

```plaintext
//MAPPER EXEC PGM=FABMMAIN
//STEPLIB DD DSN=HPS.SHPSLMDO,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
   DBD=BE1PARTS,X
   PSB=PE1CPINV,X
   PSB=PE1PPINV
   ACB=PE1CPINV
   PSB=PE1PARTS
   ACB=PE1CPPUR
   DBD=BE2PCUST
   ACB=PE1PARTS
   ACB=PE1PPINV
   DBD=B00INP01,X
   DBD=B00OUT01
/*
```

Figure 85. Example of generating DBD, PSB, ACB maps
Example: Creating a DBD and generating a DBD map

The following figure shows example JCL for running a job that creates a DBD and generates a map of the DBD.

You can use this example JCL to create or change a DBD, and obtain a map and report on it.

//DBDGEN
PROC MBR=TEMPNAME, RGN=2048K
//C EXEC PGM=IEV90, REGION=&RGN, PARM='OBJECT,NODECK'
//SYSLIB DD DSN=IMSVS.MACLIB, DISP=SHR
//SYSLIN DD UNIT=SYSDA, DISP=(PASS),
// SPACE=(80,(100,100), RLSE),
// DCB=(BLKSIZE=80, RECFM=F, LRECL=80)
//SYSPRINT DD SYSOUT=A, DCB=BLKSIZE=1089,
// SPACE=(121, (300, 300), RLSE, ROUND)
//SYSUT1 DD UNIT=SYSDA, DISP=(DELETE),
// SPACE=(1700, (100, 50))
//SYSUT2 DD UNIT=SYSDA, DISP=(DELETE),
// SPACE=(1700, (100, 50))
//SYSUT3 DD UNIT=(SYSDA, SEP=(SYSLIB, SYSUT1, SYSUT2)),
// SPACE=(1700, (100, 50))
//SYSIN DD DSNAME=DBT.SOURCE(&MBR), DISP=SHR
//L EXEC PGM=IEWL, PARM='XREF, LIST', COND=(0, LT, C), REGION=120K
//SYSLIN DD DSN=*.C.SYSLIN, DISP=(OLD, DELETE)
//SYSPRINT DD SYSOUT=A, DCB=BLKSIZE=1089,
// SPACE=(121, (90, 90), RLSE)
//SYSLMOD DD DSN=IMSVS.DBDLIB(&MBR), DISP=SHR
//SYSUT1 DD UNIT=(SYSDA, SEP=(SYSLMOD, SYSLIN)),
// SPACE=(1024, (100, 10), RLSE), DISP=(DELETE)
//*
//DBDMAP EXEC PGM=FABMDMAP, PARM=&(MBR)
//STEPLIB DD DSN=HPS.SHPSLMD0, DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB, DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//*
PEND
//*

Figure 86. Example of creating a DBD and generating a DBD map

Chapter 7. DBD/PSB/ACB Mapper utility 257
Example: Creating a PSB and generating a PSB map

The following figure shows example JCL for running a job that creates a PSB and generates a map of the PSB.

You can use this example JCL to create or change a PSB, and obtain a map and report on it.

```
//PSBGEN PROC MBR=TEMPNAME,RGN=2048K
//C EXEC PGM=IEV90,REGION=&RGN,PARM='OBJECT,NODECK'
//SYSLIB DD DSN=IMSVS.MACLIB,DISP=SHR
//SYSLIN DD UNIT=SYSDA,DISP=(,PASS),
//      SPACE=(80,(100,100),RLSE),
//      DBC=(BLKSIZE=80,RECFM=F,LRECL=80)
//SYSPRINT DD SYSOUT=A,DCB=BLKSIZE=1089,
//      SPACE=(121,(300,300),RLSE,ROUND)
//SYSPUT DD UNIT=SYSDA,DISP=(,DELETE),
//      SPACE=(1700,(100,50))
//SYSUT2 DD UNIT=SYSDA,DISP=(,DELETE),
//      SPACE=(1700,(100,50))
//SYSUT3 DD UNIT=(SYSDA,SEP=(SYSLIB,SYSTUT1,SYSTUT2)),
//      SPACE=(1700,(100,50))
//SYSSIN DD DSNNAME=OBT.SOURCE(AMBR),DISP=SHR
//L EXEC PGM=IEWL,PARM='XREF,LIST',COND=(0,LT,C),REGION=120K
//SYSLIN DD DSN=*.C.SYSLIN,DISP=(OLD,DELETE)
//SYSPRINT DD SYSOUT=A,DCB=BLKSIZE=1089,
//      SPACE=(121,(90,90),RLSE)
//SYSLMOD DD DSN=IMSVS.PSBLIB(AMBR),DISP=SHR
//SYSUT1 DD UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),
//      SPACE=(1024,(100,10),RLSE),DISP=(,DELETE)
//*
//PSBMAP EXEC PGM=FABMPMAP,PARM=(AMBR)
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//*
PEND
```

Figure 87. Example of creating a PSB and generating a PSB map
Output from the DBD/PSB/ACB Mapper utility

Output from the DBD/PSB/ACB Mapper utility consists of the SYSOUT data set and the SYSPRINT data set.

SYSOUT data set

The SYSOUT data set (activity log) contains a list of SYSIN records followed by a message for each input dbdname, psbname, and acbname.

The messages state that the DBD, PSB, or ACB:
- Has been selected
- Has been selected with an extended report
- Has not been found in the respective library

This data set also contains all error messages.

The following figure shows messages that are generated in the SYSOUT data set.

**Figure 88. Messages in the SYSOUT data set**

SYSPRINT data set

The SYSPRINT data set contains the DBD maps, PSB maps, ACB maps, and reports.

The SYSPRINT data set must contain fixed-length records of 133 bytes and a block size of 133 or a multiple of 133.

**DBD Map function**

The DBD Map function produces the DBD maps and the DBD reports. The DBD reports can be omitted.

**PSB Map function**

The PSB Map function produces a PSB Summary report for the PSB and PSB maps and PSB reports for database PCB that is defined in the PSB. The PSB reports can be omitted.

**ACB Map function**

For PSB-type ACB members, the ACB Map function produces an ACB (PSB) Summary report for the PSB-type ACB and ACB (PSB) maps, and ACB (PSB) reports for each database PCB that is defined in the ACB. For DBD-type ACB members, the ACB Map function produces an ACB (DBD) map and ACB (DBD) report.
ACB Map function does not give information about the control block in GSAM DBD and logical DBD, because those control blocks do not exist in the ACB library.

The ACB (PSB) reports and the ACB (DBD) reports can be omitted.
DBD, PSB, and ACB maps

DBD/PSB/ACB Mapper generates DBD, PSB, and ACB maps in the SYSPRINT data set.

DBD/PSB/ACB Mapper generates the maps as follows:
- A DBD map for each DBD.
- A PSB map for each database PCB within the PSB.
- An ACB (PSB) map for each database PCB within the PSB-type ACB.
- An ACB (DBD) map for each DBD-type ACB.

Subsections:
- “Format of the maps”
- “Differences among DBD, PSB, and ACB maps” on page 262
- “Example of a DBD map (single-page)” on page 263
- “Example of a DBD map (multiple-pages)” on page 263

Format of the maps

The maps produced by DBD/PSB/ACB Mapper depict the hierarchical structure of a database as described in a DBD. The map heading shows the DBD member name, volume serial number, and data set name of the library that contains the member. The creation date, time, and IMS version of the DBD are shown on the right of the data set name. It also shows the access method, or if it is a logical database LOGICAL, for the DBD currently being mapped. Both physical and logical relationships are shown. A map can be created for all full-function and Fast Path IMS database organizations except GSAM. A map cannot be created for a GSAM database, because it does not contain segments. However, a report can be created.

For physical relationships, each segment is represented by a box that contains the segment name and code. Each box (except for the root segment) is connected to its physical parent and siblings. The characters as shown in the following table are used to draw a box that shows the data set group that contains the segment. These characters are called data set group characters.

Table 20. Data set group characters

<table>
<thead>
<tr>
<th>Character</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>The first data set group</td>
</tr>
<tr>
<td>+</td>
<td>The second data set group</td>
</tr>
<tr>
<td>&quot;</td>
<td>The third data set group</td>
</tr>
<tr>
<td>.</td>
<td>The fourth data set group</td>
</tr>
<tr>
<td>=</td>
<td>The fifth data set group</td>
</tr>
<tr>
<td>-</td>
<td>The sixth data set group</td>
</tr>
<tr>
<td>#</td>
<td>The seventh data set group</td>
</tr>
<tr>
<td>%</td>
<td>The eighth data set group</td>
</tr>
<tr>
<td>;</td>
<td>The ninth data set group</td>
</tr>
<tr>
<td>’</td>
<td>The 10th data set group</td>
</tr>
<tr>
<td>V</td>
<td>Virtual logical child</td>
</tr>
<tr>
<td>C</td>
<td>Pointer and parent segment concatenation</td>
</tr>
</tbody>
</table>
Data set group characters are used to express up to the maximum number of data set groups allowed for a database. A DEDB database can have up to 2048 areas. The area segments use the same characters as the data set groups. For more than 10 areas, the characters are simply repeated; that is, 11 through 20 use the same characters as 1 through 10, 21 through 30 use the same characters as 1 through 10, and so on.

VAR in the top line of a box indicates a variable-length segment. SXD in the top line of a box indicates that the segment has secondary index fields. If the segment is a sequential dependent segment (SDEP), SDEP is shown in the upper-right corner of the box. The segment code is placed in the bottom line of each box.

MULT in the top line of a box indicates that the segment has multiple secondary indexes.

Logical relationships for the segment box are indicated by the segment name of the logical parent or logical child and, where necessary, the name of the database that contains that segment. Dependent logical segments are not connected as physical segments are; rather, they appear first under their associated segment in a vertical row, one under another.

If the map is too wide or long to fit on a single page, the map is split and printed on as many contiguous pages as needed.

**Differences among DBD, PSB, and ACB maps**

Although the format of the four types of maps is similar, certain elements are different. The following list summarizes the differences between the DBD map and other types of maps:

**Differences in PSB maps**

PSB maps are similar to the DBD maps that the DBD Map function generates. The following fields are different in DBD maps and PSB maps:

- **DBDMAP OF dbdname in PSB-psbname** is shown in the header.
- 2ND is shown in the upper-right corner of the box when the segment contains the secondary index specified in PROCSEQ of DB PCB in the PSB map.
- SXD is not shown in the PSB map.
- The processing options (PROCLOPT values) used for the segment are shown at the lower-left corner of the box.

A special case is introduced, however, in which the list of segments included in the map is customized to include only those called for in the database PCB sensitive segment table. Also, if the sensitive segment table and the database PCB attributes call for inversion of the database because of a secondary index processing sequence, this is simulated before the map is prepared. The order of the segments in the map is the same as the order of the SENSEG statements in the PSB.

**Differences in ACB (PSB) maps**

ACB (PSB) maps are similar to the PSB maps that are generated by the PSB Map function. The following fields are different between PSB maps and ACB (PSB) maps:

- If the sensitive segment is a virtual child segment, information about this segment is not shown in the ACB (PSB) map.
- 2ND and SXD are not shown in the ACB (PSB) map.
Differences in ACB (DBD) maps

ACB (DBD) maps are similar to the DBD maps that are generated by the DBD Map function.

Example of a DBD map (single-page)

The following figure shows an example of the DBD map that fits in a single page.

---

Figure 89. DBD map (single-page)

Example of a DBD map (multiple-pages)

The following figure shows an example of the DBD map that spans across pages.
ALL SEGMENTS WITHIN A SINGLE DATA SET GROUP ARE DISPLAYED USING THE SAME CHARACTER FOR BOX DELIMITERS.

Figure 90. DBD map (multiple-pages) (Part 1 of 2)
PSB and ACB summary reports

The PSB Summary report and the ACB (PSB) Summary report provide overall information about the PSB or the PSB-type ACB, its attributes, and the PCBs it contains. These reports are generated in the SYSPRINT data set.

Up to 2500 PCBs can be printed on the PCB number column of the PSB Summary report.

The header of the PSB Summary report contains the following common items:

- PSB member name
- Volume serial number and data set name of the library containing the PSB member
- Date, time, and IMS version when the PSB was generated

Subsections:

- “Sample report” on page 266
- “Report field description” on page 267
Sample report

The following figures show an example of the PSB Summary report.

Figure 92. PSB Summary report (Part 1 of 2)
Report field description

This report has seven sections:

PSB Prefix Summary
A list of the PSB attributes, such as I/O area size and SSA size.

TP PCB Summary
A list of the associated I/O PCBs.

DB PCB Summary
A list of the associated database PCBs.

GSAM PCB Summary
A list of the associated GSAM PCBs.

TP PCB Detail
List of associated I/O PCBs, with the attributes of each of these PCBs. The PCB name and the LIST parameter option are printed, if they exist.

DB PCB Detail
List of associated database PCBs, with the attributes of each and a table of the sensitive segments for each. Sensitive field information is also displayed whenever it is contained in a DB PCB. If any subset pointers are defined in a PCB for a DEDB database, this information is printed on the report. The PCB name and the LIST parameter option are printed, if they exist.

For a PCB for a DEDB database, MULTIPLE POSITIONING REQUESTED is always printed regardless of the value specified for the POS parameter of the PCB statement.

GSAM PCB Detail
List of associated GSAM PCBs with the attributes of each of these PCBs.

Most of the fields in the PSB Summary report are self-explanatory. The sensitive segment fields contain:

- The segment name
- The name of the parent segment
- The processing options for the segment

For ACB (PSB) Summary reports, if the sensitive segment is a virtual logical child segment, VIRTUAL is shown on the right side of the segments processing option.
The sensitive field lines contain:

- The field name
- The starting position of the field within its segment
- Whether it can be altered during a replace call (shown by REPLACE=YES or REPLACE=NO)

**DBD, PSB, and ACB reports**

DBD/PSB/ACB Mapper generates DBD, PSB, and ACB reports in the SYSPRINT data set. Each report contains details about the DBD, PSB, PSB-type ACB, or DBD-type ACB.

DBD/PSB/ACB Mapper generates the reports as follows:

- If the control statement for a DBD contains only the *dbdname*, a DBD report is generated.
- If the control statement for a PSB contains only the *psbname*, a PSB report is generated for each database PCB within the PSB.
- If the control statement for a PSB-type ACB contains only the *acbname*, an ACB (PSB) report is generated for each database PCB within the PSB-type ACB.
- If the control statement for a DBD-type ACB contains only the *acbname*, an ACB (DBD) report is generated for the DBD-type ACB.

The DBD, PSB, ACB reports show the following information:

- Database information
- Data set group or DEDB area information
- Segment information
- Field information

**Subsections:**

- “Differences between DBD, PSB, and ACB reports”  
- “Example of the DBD report” on page 269  
- “Example of the PSB report” on page 271  
- “Report field description for the database information part” on page 272  
- “Report field description for the data set group or DEDB area information part” on page 274  
- “Report field description for the segment information part” on page 274  
- “Report field description for the field information part” on page 276

**Differences between DBD, PSB, and ACB reports**

Although the formats of the reports are similar between the four types of reports, certain fields are different. The following list summarizes the differences between the DBD report and other types of reports.

**Differences in PSB reports**

PSB reports are similar to the DBD reports that are generated by the DBD Map function.

A special case is introduced, however, in which the list of segments included in the report is customized to include only those called for in the database PCB sensitive segment table. Also, if the sensitive segment table and the database PCB attributes call for inversion of the database because of a secondary index processing sequence, this is simulated before the
The order of the segments in the report is the same as the order of the SENSEG statements in the PSB.

Differences in ACB (PSB) reports
ACB (PSB) reports are similar to the PSB reports that are generated by the PSB Map function. If the ACB control block does not contain information that is valid for the report field, N/A is printed in the field of the ACB (PSB) report. The following fields are different between PSB reports and ACB (PSB) reports:

- If the sensitive segment is a virtual child segment, information about this segment is not shown on the ACB (PSB) report.
- *INSENS* shown in the *SRCSEG* line means that the index source segment name cannot be obtained from the ACB because the segment is insensitive in the PSB.

Differences in ACB (DBD) reports
ACB (DBD) reports are similar to the DBD reports that are generated by the DBD Map function. If the ACB control block does not contain information that is valid for the report field, N/A is printed in the field of the ACB (DBD) report.

- When DBD/PSB/ACB Mapper cannot obtain complete segment name information from one or more ACBs, the name is shown in the ACB (DBD) map and the ACB (DBD) report as follows:
  
  $FABMnnn \ (nnn \ \text{is \ the \ segment \ code \ in \ a \ DBD})$

- Because DBD/PSB/ACB Mapper cannot obtain paired segment name information for virtually paired logical relationship, the name is not shown in the ACB (DBD) report.
- Because DBD/PSB/ACB Mapper cannot obtain segment name information and database name information of the SOURCE parameter for virtually paired logical relationship, these names are shown in the ACB (DBD) report as follows:
  
  - $SEGMMnnn \ (nnn \ \text{is \ a \ sequential \ number \ in \ a \ DBD})$
  - $DBDMnnn$

Then the program writes message FABM0043W to the SYSOUT data set.

Example of the DBD report
The following figures show an example of the DBD report.
**Figure 94. DBD report (Part 1 of 2)**

```
<table>
<thead>
<tr>
<th>DDDD</th>
<th>SSS</th>
<th>CCC</th>
<th>RRRR</th>
<th>SSS</th>
<th>DDDD</th>
<th>V</th>
<th>V</th>
<th>N</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>D</td>
<td>S</td>
<td>C</td>
<td>R</td>
<td>R</td>
<td>S</td>
<td>D</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>D</td>
<td>D</td>
<td>S</td>
<td>C</td>
<td>R</td>
<td>R</td>
<td>S</td>
<td>D</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>D</td>
<td>D</td>
<td>S</td>
<td>S</td>
<td>C</td>
<td>R</td>
<td>R</td>
<td>S</td>
<td>D</td>
<td>V</td>
</tr>
<tr>
<td>D</td>
<td>D</td>
<td>S</td>
<td>S</td>
<td>C</td>
<td>R</td>
<td>R</td>
<td>S</td>
<td>D</td>
<td>V</td>
</tr>
</tbody>
</table>

**Version=08/10/15**

**Level= 3**

**Segments= 8**

**Data Set Groups= 2**

**Randomizing Routine=DFSHDC40**

**Root Anchor Points= 2**

**Max RBN= 20**

**Bytes= Unlimited**

**Id= 1**

**DSCRSDV0**

**Prime DS Log. Rcd. Len=** 2041

**Blocksize=** 2048

**Segment Length Max=** 80

**Min=** 42

**Key Length Max=** 20

**Seg-Name= Phys**

**D-B-Name= Form**

**Log Child**

**Seg-Name** | **Sc# | Par | Len---Freq--- | **T** | **P** | **T** | **P** | **H** | C | C | C | C | P | N-Seq | **OR** | **OR** | **Insrt** | Fld-Name | Len | Strt | Pnter | Rules
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SSCRSP00+</td>
<td>1 1 0 50</td>
<td>0.00</td>
<td>X</td>
<td>X</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>L</td>
<td>V</td>
<td>V</td>
<td>LAST</td>
<td><em>Pfx</em></td>
<td>Len= 30</td>
<td>Max= 50</td>
<td>Pfx+Max= 80</td>
<td>MIN= 22</td>
<td>Pfx+Min= 52</td>
<td><em>LC</em></td>
<td>SSSCHP13</td>
<td>DSSCHVYN</td>
<td>SNGL</td>
<td>HERE</td>
<td><em>LC</em></td>
</tr>
</tbody>
</table>
**Figure 95. DBD report (Part 2 of 2)**

**Example of the PSB report**

The following figure shows an example of the PSB report.
Report field description for the database information part

The database information part includes the following information:

- On the first two lines of the report (underneath the heading):
  - DBD member name
– Volume serial number and data set name of the library that contains the DBD member
– Date and time when the DBD was generated
– Database name
– The IMS database organization and access method, or for a logical database, LOGICAL

• The database name, in large letters, on the next several lines.
• Additional database information is printed on the next several lines:
  – INDEX-SYS-DATA-PROTECTED means that the database is a secondary index and is data-protected.
  – DOS INDEX COMPAT means that the database is an index and was created by use of DLI/DOS.
  – PASSWORD-PROTECTED means that the database is password-protected (VSAM only).
  – USER EXIT DFSDBUX1 SPECIFIED means that user exit DFSDBUX1 is called at the beginning and end of each database call.
  – The user-specified version information or time stamp value.
  – For a DBD-type ACB for a DEDB, this value is obtained from the PSB-type ACB that references the DBD. If a problem occurs when reading the PSB-type ACB, this field is blank.
  – The number of levels in the database.
  – The number of segment types in the database.
  – The number of data set groups, or DEDB areas.

• If the exit parameter is specified on the DBD statement, the following information is printed:
  – The exit name
  – KEY or NOKEY
  – DATA or NODATA
  – PATH or NOPATH
  – CASCADE or NOCASCADE
  – LOG or NOLOG or none
  – DLET or NODLET or none
  – BEFORE or NOBEFORE or none

**Note:** If the exit parameter is specified on the DBD statement, it is applied to all segments within the physical database structure. To override the specification or to limit the specification to certain segments, the exit parameter is specified on the SEGM statement. So if the EXIT=NONE parameter is specified on all SEGM statements, the exit parameter on the DBD statement is not printed, because it has no meaning.

If the CASCADE option is specified, the following information is printed next to the CASCADE:
– KEY or NOKEY
– DATA or NODATA
– PATH or NOPATH

• If the database organization is HDAM or DEDB, the following information is printed:
  – The name of the randomizing routine
  – The number of root anchor points
  – For HDAM only, the maximum number of relative blocks
– For HDAM only, the maximum number of bytes that can be placed in the root-addressable area

• If the database organization is GSAM, the organization and block information are printed on the line following the level, segments, and data set group information. The report is then terminated, because no further information is available for GSAM.

• If the database organization is MSDB, the terminal-related codes are printed.

• If there are shared index segments in the database, the names of the sharing databases are printed.

**Report field description for the data set group or DEDB area information part**

The following information is printed for each data set group or DEDB area in the database:

• The identification (ID) number.
  – For a data set group or a DEDB area, the identification (ID) number with the data set group character is printed.

• The data set group name.

• Information about the prime data set: the logical record length, block size, maximum and minimum segment lengths, and device type.

• Information about the overflow data set: logical record length and block size. Also shown on this line are the maximum and minimum key lengths used, the total number of segment types in the group, and the number of virtual segments, if any.

• For DEDB areas, the unit of work and the root numbers are printed.

• For HDAM, HIDAM, PHDAM, or PHIDAM databases, the free block frequency factor and the free space percentage factor numbers, if any, are printed.

**Report field description for the segment information part**

The next lines in the report consist of all data concerning each segment type and the fields within them.

• The segment name, with the data set group character. If the segment is virtual, a V is printed alongside the segment name. If the segment contains concatenated keys, a C is printed alongside the segment name.

• The segment code number.

• The database level the segment is in.

• The segment code of the segment's physical parent.

• The segment frequency number. If there is none, a zero (0) is printed.

• The types of pointers within the segment, if any, are specified by an X under each pointer ID:
  - CTR: counter field
  - PTF: physical twin forward pointer
  - PTB: physical twin backward pointer
  - PP: physical parent pointer
  - LTF: logical twin forward pointer
  - LTB: logical twin backward pointer
  - LP: logical parent pointer
  - PHF: hierarchical forward pointer
  - PHB: hierarchical backward pointer
  - EPS: extended pointer set
The following information shows the number of each type of child pointer (if any) for the segment:

- **PCF** physical child first count
- **PCL** physical child last count
- **LCF** logical child first count
- **LCL** logical child last count

- **The next section under the heading RULES contains the rules used for inserting, deleting, and replacing occurrences of the segment type, as follows:**
  - **P** A physical path must be used to insert, delete, or replace.
  - **V** A virtual path must be used to insert, delete, or replace.
  - **L** A logical path must be used to insert, delete, or replace.
  - **B** Used only for delete; means a bidirectional virtual path.

If the report is for MSDB database, N/A is issued in this field on DBD report because RULES is not supported for MSDB.

- **The next column, headed PHYS. N-SEQ INSRT, shows where the new segment occurrences are inserted into the physical database that establishes the physical twin sequence by FIRST, LAST, or HERE. The value is used only when segments are processed with no sequence field or with a non-unique sequence field.**

If the report is for an MSDB database, N/A is printed in this field on DBD report because RULES is not supported for MSDB.

If the segment type has a field that is used for secondary indexing, **SNDIXD** is printed following INSERT RULES.

If the segment is a sequential dependent segment (SDEP), **SDEP** is printed following INSERT RULES.

The segment length type is the last item printed; either **FIX LEN** or **VAR LEN**.

If the segment has a compression routine, its name is printed. **KEY-EP** means that key compression is allowed. **INIT-EP** means that initialization and termination processing are required by the compression routine.

If the exit parameter is specified on the DBD statement or the SEGM statement, one of the following is printed next to the **CHANGED DATA EXIT**:

- **SAME AS DBD** means that there is no exit parameter on the SEGM statement and the exit parameter on the DBD statement will apply to this statement too.
- **NONE** means that the EXIT=NULL is specified on the SEGM statement and the exit parameter on the DBD statement will not apply to this statement.
- **EXIT=** means that the exit parameter on the DBD statement will be overridden by the exit parameter on this SEGM statement. The value can be the following operands:
  - The exit name
  - **KEY** or **NOKEY**
  - **DATA** or **NODATA**
  - **PATH** or **NOPATH**

The following information is printed on the next line:

- **CASCADE** or **NOCASCADE**

  If the **CASCADE** option is specified, the following information is printed next to the **CASCADE**:
  - **KEY** or **NOKEY**
  - **DATA** or **NODATA**
  - **PATH** or **NOPATH**
  - **DLET** or **NODLET** or **none**
  - **BEFORE** or **NOBEFORE** or **none**
and the following information is printed on the next line:

– LOG or NOLOG

For DEDB DBDs, the next line under the segment name shows how many subset pointers, if any, have been defined for this segment.

**Report field description for the field information part**

The next line on the right side describes the prefix information. This contains the following data:

- *PFX* identifies the line as prefix data.
- LEN shows the length of the data portion of the segment.
- PFX+MAX shows the total maximum length of the segment data and prefix.

If the report is for a DEDB database, N/A is issued following *PFX* because DBD has no valid segment length information for the DEDB database.

For a variable-length segment, the following information is printed on the next line:

- MIN shows the smallest length of the data portion that is used by IMS.
- PFX+MIN shows the minimum length that is used by IMS for a segment.

The following lines for the segment specify segment type, segment field, or both:

- If the identification is for a segment, the lines contain:
  - The segment type:
    - *PR* describes a paired segment
    - *LC* describes a logical child
    - *LP* describes a logical parent
  - The segment name
  - The name of the database that contains this segment

    If this segment has multiple secondary indexes, names of all the index databases are printed.

  - The type of pointer:
    - INDX index pointer
    - SNGL contains a physical child first pointer
    - DBLE contains both a physical child first and a physical child last pointer
    - NONE contains no pointer
    - PHYS contains a physical parent pointer
    - VIRT contains a virtual parent pointer
  - The next field that contains the type of insertion rule: FIRST, LAST, or HERE.

  - RKSIZE shows the length of the root key of the target segment.

- If the line is for a defined field of the segment, the line consists of:
  - *FLD* specifies defined field
  - Name field name
  - Length field length
  - Start starting position of field within segment
  - Type defines the type of field:
    - CHARACTER
    - HEXADECIMAL
    - PACKED
    - HALFWORD
If the field is a sequence field for the segment, the following is printed:
- SEQ denotes the field as a sequence field.
- UNIQUE or MULTIPLE specifies whether the sequence field is to have different values in all occurrences of the segment, or can have duplicates of the same values.

If the line is for an indexed field, the fields are:
- *XFDM* indexed field
- Field name self-explanatory
- SECONDARY INDEXED FIELD self-explanatory

Following the XFDM line could be the following:
- SUBSEQUENCE shows that there are subsequence fields.
- SYMBOLIC shows that the pointers are symbolic.
- **SRCSEG** is printed along with the segment name. If the source segment is same as the target segment, **SAME** is printed.
- CONSTANT=X'xx' is printed if a constant xx is specified in the definition of the XDFLD statement.
- *NULLEXIT* is printed if a null value or an exit routine has been specified in the XDFLD statement:
  - The null value is printed as NULL=X'xx'.
  - The exit routine is printed as EXIT-name.
- **PSELRTN** is printed if PSELRTN= is specified in the XDFLD statement. PSELOPT shows the partition selection option. The default value for PSELOPT is MULT.
- For other index field types, the following is printed:
  - **SEARCH**
  - **SUBSEQ**
  - **SOURCE**
  - The field name, length, and start values. The field name shows */CK* if the SRCH= or SUBSEQ= parameter of the XDFLD statement specifies its operand in the following format: */CK, start_value, length_value*
  - If there is a source segment name for the XDFLD, *FSRCSEG* and the segment name.
  - If there is a source segment for the index segment, the phrase *SRC* is printed along with the source segment name and the name of the database in which the source segment resides. Along with this, one of the following is printed, depending on what part of the source segment is used:
    - DATA
    - KEY
    - PATH
Chapter 8. DBD/PSB/ACB Reversal utility

The DBD/PSB/ACB Reversal utility converts the DBD, PSB, and ACB control blocks back into IMS DBDGEN or IMS PSBGEN utility control statements.

Topics:
- “DBD/PSB/ACB Reversal utility overview” on page 280
- “Restrictions for the DBD/PSB/ACB Reversal utility” on page 283
- “Converting IMS control blocks to control statements” on page 284
- “JCL requirements for the DBD/PSB/ACB Reversal utility” on page 285
- “Control statements for the DBD/PSB/ACB Reversal utility” on page 288
- “JCL examples for the DBD/PSB/ACB Reversal utility” on page 300
- “Output from the DBD/PSB/ACB Reversal utility” on page 304
- “DBD/PSB/ACB Reversal Site Default Generation utility” on page 326
The DBD/PSB/ACB Reversal utility converts DBD, PSB, and ACB control blocks back into IMS DBDGEN and IMS PSBGEN utility control statements.

Subsections:
- “Function overview”
- “Program structure” on page 281
- “Data flow” on page 281

Function overview

The utility provides the following functions:

**DBD Reversal**
This function reads one or more DBDs from a DBD load library and converts them back to IMS DBDGEN utility control statements.

**PSB Reversal**
This function reads one or more PSBs from a PSB load library and converts them back to IMS PSBGEN utility control statements.

**ACB Reversal**
This function reads one or more ACBs from an ACB load library and converts them back to IMS PSBGEN and DBDGEN utility control statements. Depending on the specified operand, it processes both PSB-type and DBD-type ACBs, or only DBD-type ACBs.

**Site Default Generation utility**
You can use the Reversal Site Default Generation utility to set your own default values for the Reversal SYSIN control statements.

The DBD/PSB/ACB Reversal program also generates the following reports. These reports represent the information about IMS DBD, PSB, and ACB libraries such as DBD/PSB/ACB organization, PCB PROCOPT, and relation among members.
- DBD Cross-Reference report
- ACB(DBD) Cross-Reference report
- ACB(PSB) Cross-Reference report
- PCB PROCOPT report
- PCB/ACB(PSB) PROCOPT report
- DBD to DBD cross-reference report
- PSB to DBD cross-reference report
- ACB(DBD) to ACB(DBD) cross-reference report
- ACB(PSB) to ACB(DBD) cross-reference report
- DBD cross-reference by DDname report
- ACB(DBD) cross-reference by DDname report
- DBD segment reference report
- PSB segment reference report
- PSB PROCOPT reference report
- ACB(PSB) PROCOPT reference report
- Unreferenced ACB(DBD) report
- DBD library member list report
- PSB library member list report
You can optionally request the DBD/PSB/ACB Reversal program to generate records suitable for the input to the DBD/PSB/ACB Mapper program by specifying the MAPOUT DD statement.

Related reading:
- For details about each report, see “SYSPRINT data set” on page 308.
- For details about creating the input for the DBD/PSB/ACB Mapper program, see the explanation of the MAPOUT DD statement in “JCL requirements for the DBD/PSB/ACB Reversal utility” on page 285.

Program structure

The DBD/PSB/ACB Reversal utility provides the following programs:

The FABNRVRS program
This program controls other load modules and converts IMS control blocks (DBDs, PSBs, and ACBs) back into IMS DBDGEN/PSBGEN utility control statements. Based on the user specification, the program also generates various summary reports about IMS libraries. This module uses a simple input format that is specified in the SYSIN data set.

The FABNTGEN program
This program is the Site Default Generation utility for DBD/PSB/ACB Reversal. The program creates a user site default table for the FABNRVRS SYSIN statement. It can also report values that are registered in the SYSIN site default table.

Data flow

The general data flow for the DBD/PSB/ACB Reversal utility (FABNRVRS) and the site default generation utility (FABNTGEN) is shown in the following figure. The input for the DBD/PSB/ACB Reversal utility is the SYSIN data set and the DBDLIB, PSBLIB, or ACBLIB for which sources are output and reports are created, and the output is DBDGEN/PSBGEN control statements and DBD/PSB/ACB Mapper control statements, reports, and activity log.
Figure 98. Data flow for DBD/PSB/ACB Reversal
Certain restrictions apply when you use the DBD/PSB/ACB Reversal utility.

DBD/PSB/ACB Reversal re-creates the DBD/PSB source to be used for IMS of the same version and release by which the DBD/PSB were generated or for higher version of IMS. If a DBD/ACB member for HALDB is used as input data, the following reports are not generated, because the DBD/ACB library contains no information about the DD name:
- DBD XREF by DDname report
- ACB(DBD) XREF by DDname report

If the DBD member is specified to generate the reports, message FABN0054I is issued.

DBD/PSB/ACB Reversal cannot create reports if an alias name that is defined to the database is specified on the following control statements:
- XREF
- DDNAMES
- SEGREF

For the restrictions that apply to the generated control statements, see “Restrictions on the generated control statements” on page 306.
Converting IMS control blocks to control statements

To convert DBD/PSB/ACB control blocks back into IMS DBDGEN/PSBGEN utility control statements by using the DBD/PSB/ACB Reversal utility, you must prepare the JCL for the DBD/PSB/ACB Reversal utility and submit the job.

About this task

Sample JCL for the DBD/PSB/ACB Reversal utility is in the SHPSJCL0 library, member FABLIVP1. You can modify this sample JCL and then use it to run the utility.

Procedure

1. In the DBD/PSB/ACB Reversal JCL, code the EXEC statement and DD statements.
   You can use the Reversal Site Default Generation utility to set your own default values for the SYSIN control statements.
   To create a DBD/PSB/ACB Mapper input, specify the MAPOUT DD statement.
   See the following topics for additional information:
   - For the format of the EXEC statement and the list of DD statements, see “JCL requirements for the DBD/PSB/ACB Reversal utility” on page 285.
   - For a description about the Reversal Site Default Generation utility, see “Reversal Site Default Generation utility overview” on page 326.
   - For an instruction to use the Reversal Site Default Generation utility, see “Setting site default values for the DBD/PSB/ACB Reversal utility” on page 326.

2. In the SYSIN data set, code the control statements for the DBD/PSB/ACB Reversal utility.
   See “Control statements for the DBD/PSB/ACB Reversal utility” on page 288.

3. Submit the job.

4. Check the output data sets that are generated.
   See “Output from the DBD/PSB/ACB Reversal utility” on page 304.
JCL requirements for the DBD/PSB/ACB Reversal utility

When you code the JCL to run the DBD/PSB/ACB Reversal utility, include the EXEC statement and appropriate DD statements.

Subsections:
* “JCL example”
* “EXEC statement”
* “DD statements”

JCL example

An example of the JCL that is required for DBD/PSB/ACB Reversal is shown in the following figure.

```c
//stepname  EXEC  PGM=FABNRVRS
//STEPLIB  DD  DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB  DD  DSN=IMSVS.DBDLIB,DISP=SHR
//PSBLIB  DD  DSN=IMSVS.PSBLIB,DISP=SHR
//ACBLIB  DD  DSN=IMSVS.ACBLIB,DISP=SHR
//SYSOUT  DD  SYSOUT=A
//SYSPRINT  DD  SYSOUT=A
//SYSPUNCH  DD  SYSOUT=B,FREE=CLOSE
//DBDSRC  DD  DSN=PDS.DBDSRC,DISP=SHR
//PSBSRC  DD  DSN=PDS.PSBSRC,DISP=SHR
//MAPOUT  DD  SYSOUT=B,FREE=CLOSE
//SYSIN  DD  *

(*control statements*)
/*
```

Figure 99. Example of DBD/PSB/ACB Reversal JCL (FABNRVRS JCL)

EXEC statement

This statement must be in the following format:

```c
//stepname  EXEC  PGM=FABNRVRS
```

DD statements

Code the following DD statements to identify the source of input and the placement of output information:

**STEPLIB DD or JOBLIB DD**

This DD statement is required. This input DD statement defines the IMS Library Integrity Utilities load module library.

**DBDLIB DD**

This statement is required when you process DBDs. This statement is also required when you specify the OPTION ACB_GSAM=YES option to process ACBs that have GSAM information.

The DBDLIB DD input data set is the load library that contains the DBDs for which reports and control statements are created.

**PSBLIB DD**

This statement is required when you process PSBs. This statement is also required when you specify the OPTION ACB_GSAM=YES option to process ACBs that have GSAM information.
The PSBLIB DD input data set is the load library that contains the PSBs for which reports and control statements are created.

**ACBLIB DD**
This statement is required when ACB operations are to be performed.
The ACBLIB DD input data set is the load library that contains the PSB-type ACBs and the DBD-type ACBs for which reports and control statements are created.

**SYSOUT DD**
This DD statement is required. The SYSOUT DD data set contains all activity messages and error messages.
The record format is fixed-blocked, and the logical record length is 133.
The block size, if coded, must be a multiple of 133.

**SYSPRINT DD**
This DD statement is not required if only the DECODE keyword is specified in the SYSIN data set; otherwise it is required.
The SYSPRINT DD data set contains various reports that correspond to the SYSIN control statements. The record format is fixed-blocked, and the logical record length is 133. The block size, if coded, must be a multiple of 133.

**SYSPUNCH DD**
This optional data set contains the IMS DBDGEN/PSBGEN utility control statements re-created by DBD/PSB/ACB Reversal function.
The record format is fixed-blocked, and the logical record length is 80. The block size, if coded, must be a multiple of 80.

**DBDSRC DD**
This optional PDS or PDSE contains the IMS DBDGEN utility control statements re-created by DBD or ACB reversal function.
The record format is fixed-blocked, and the logical record length is 80. The block size, if coded, must be a multiple of 80.

**PSBSRC DD**
This optional PDS or PDSE contains the IMS PSBGEN utility control statements re-created by the PSB or ACB Reversal function.
The record format is fixed-blocked, and the logical record length is 80. The block size, if coded, must be a multiple of 80.

**Note:** If the keyword DECODE is specified, you must specify either of the following data sets as the output data set:
- SYSPUNCH DD
- DBDSRC DD, PSBSRC DD, or both

**Related reading:** For more information about the relation of DD statements and control statements, see [Table 21 on page 298](#).

**MAPOUT DD**
This data set contains input to the DBD/PSB/ACB Mapper program. It is an optional output data set for the DBD/PSB/ACB Reversal function. If it is specified, the input to the DBD/PSB/ACB Mapper is generated.
The record format is fixed-blocked and the logical record length is 80. The block size, if coded, must be a multiple of 80.
SYSIN DD
This DD statement is required. The input data set, SYSIN DD, contains the control statements for DBD/PSB/ACB Reversal program.

The record format is fixed-blocked, and the logical record length is 80. The block size, if coded, must be a multiple of 80.

**Related reading:** For the format of the control statements, see “Control statements for the DBD/PSB/ACB Reversal utility” on page 288.

OPTPRT DD
This DD statement is optional. The OPTPRT DD data set contains the Run-time Option report. The record format is fixed-blocked, and the logical record length is 133. The block size, if coded, must be a multiple of 133.
Control statements for the DBD/PSB/ACB Reversal utility

The input for the DBD/PSB/ACB Reversal utility consists of control statements in the SYSIN data set. These control statements specify the functions to be performed and the runtime options.

The DBD/PSB/ACB Reversal utility supports two types of control statements:

**Runtime option control statements**
This type of statement specifies the runtime options. A runtime option control statement consists of a runtime option keyword and its associated options.

**Function control statements**
This type of statement specifies the function to be performed. A function control statement consists of a function keyword, a function keyword operand, and a function keyword option.

Subsections:
- “Control statement example”
- “Syntax rules” on page 290
- “Runtime option control statements” on page 290
- “Function control statements” on page 294
- “Function keyword operands” on page 295
- “Function keyword options” on page 295
- “Quick reference for control statements and DD statements” on page 298

**Control statement example**

The following figure shows an example of the control statements for DBD/PSB/ACB Reversal.

In this example:
- The first line specifies the SYSIN DD.
- The second and third lines specify runtime option control statements.
  - ACB_GSAM=YES is applied to all subsequent function control statements except the statements that contain a SEGREF function keyword.
  - PGM_COBOL=YES is applied to all subsequent function control statements that contain a DECODE or LIST function control keyword.
  - ACB_REFERENCED=YES is applied to all subsequent function control statements that contain a DECODE, LIST, DDNAMES, or XREF function control keyword.
  - COMMENT=YES, COMPRESS=YES, PCB_LABEL=YES, and SENSEG_PROCOPT=YES are applied to all subsequent function control statements that contain a DECODE function control keyword.
- The fourth and subsequent lines specify function control statements.
//SYSIN DD *
OPTION ACB_GSAM=YES, PGM_COBOL=YES, ACB_REFERENCED=YES
DECOPT COMMENT=YES, COMPRESS=YES, PCB_LABEL=YES, SENSEG_PROCOPT=YES
DECODE DBD ALL
DECODE DBD INCLUDE=member, member,...
DECODE DBD EXCLUDE=member, member,...
DECODE PSB ALL
DECODE PSB INCLUDE=member, member,...
DECODE PSB EXCLUDE=member, member,...
DECODE ACB ALL
DECODE ACB INCLUDE=member, member,...
DECODE ACB EXCLUDE=member, member,...
DECODE ACBPDBD ALL
DECODE ACBPDBD INCLUDE=member, member,...
DECODE ACBPDBD EXCLUDE=member, member,...
LIST DBD ALL
LIST DBD INCLUDE=member, member,...
LIST DBD EXCLUDE=member, member,...
LIST PSB ALL
LIST PSB INCLUDE=member, member,...
LIST PSB EXCLUDE=member, member,...
LIST ACB ALL
LIST ACB INCLUDE=member, member,...
LIST ACB EXCLUDE=member, member,...
DDNAMES DBD ALL
DDNAMES DBD INCLUDE=member, member,...
DDNAMES DBD EXCLUDE=member, member,...
DDNAMES ACB ALL
DDNAMES ACB INCLUDE=member, member,...
DDNAMES ACB EXCLUDE=member, member,...
PROCOPT PSB ALL
PROCOPT PSB INCLUDE=member, member,...
PROCOPT PSB EXCLUDE=member, member,...
PROCOPT ACB ALL
PROCOPT ACB INCLUDE=member, member,...
PROCOPT ACB EXCLUDE=member, member,...
XREF DBD ALL
XREF DBD INCLUDE=member, member,...
XREF DBD EXCLUDE=member, member,...
XREF PSB ALL
XREF PSB INCLUDE=member, member,...
XREF PSB EXCLUDE=member, member,...
XREF ACB ALL
XREF ACB INCLUDE=member, member,...
XREF ACB EXCLUDE=member, member,...
XREF PSB ALL DBDNAME=member
XREF ACB ALL DBDNAME=member

Figure 100. Control statements for DBD/PSB/ACB Reversal (Part 1 of 2)
Syntax rules

The control statements must adhere to the following syntax rules:

• Control statements can start anywhere after the second column.

• A statement with an asterisk (*) in column 1 is treated as a comment.

• The comment line, which has an asterisk (*) in column 1, is allowable between continuous lines.

• Member names in a statement must be separated by commas and must end with a blank. If a comma is used instead of a blank, the processing is continued to the next line.

Runtime option control statements

Runtime option control statements control the behavior of the subsequent function control statements. When no function control statements follow a runtime option control statement, the runtime option control statement is not used.

The following keywords are available (the abbreviations that can be used are shown in parentheses):

OPTION
This keyword affects all subsequent function control statements. This keyword is optional. If specified, it must be placed before any other control statements (blank and comment lines can precede the OPTION keyword), and must be followed by the following keyword:

ACB_GSAM=

Specifies whether to obtain and use GSAM information when decoding or reporting ACBs.

YES DBD/PSB/ACB Reversal obtains GSAM information from the DBD and PSB libraries when processing ACBs, and uses the information to decode or to report the ACBs. ACB_GSAM=YES affects the following function control statements:

• DECODE ACB
• DECODE ACBPSB
• LIST ACB
• DDNAMES ACB
• PROCOPT ACB
• XREF ACB
• POPTREF ACB

To specify ACB_GSAM=YES, the following conditions must be met:
• ACBLIB DD, DBDLIB DD, and PSBLIB DD must be specified in the JCL.
• The DBD or the PSB must be consistent with the ACB.

NO DBD/PSB/ACB Reversal does not use GSAM information to process ACBs. ACB_GSAM=NO is the default value.

ACB_REFERENCED=
Specifies whether to process all DBD-type ACBs including those that are not referenced by any PSB-type ACBs.

When you specify ACB_ALL, the utility processes all PSB-type ACBs and all DBD-type ACBs that the PSB-type ACBs refer to. If you also specify ACB_REFERENCED=NO, the utility expands the scope and additionally processes DBD-type ACBs that are not referred to by PSB-type ACBs.

YES Decodes or reports PSB-type ACBs and DBD-type ACBs that the PSB-type ACBs refer to. ACB_REFERENCED=YES is the default value.

NO Decodes or reports all PSB-type ACBs and DBD-type ACBs, including DBD-type ACBs that are not referred to by PSB-type ACBs.

ACB_REFERENCED affects the following function control statements:
• DECODE ACB ALL
• LIST ACB ALL
• DDNAMES ACB ALL
• XREF ACB ALL

PGM_COBOL=
Specifies whether to print LANG=COBOL or LANG=ASSEM in the decoded IMS PSBGEN utility control statements.

If a PSB is generated with LANG=COBOL or LANG=ASSEM, the DBD/PSB/ACB Reversal utility cannot identify the original LANG value because the PSB does not contain the original LANG value. Use the PGM_COBOL keyword to select the LANG value that meets your needs.

YES Prints LANG=COBOL in the IMS PSBGEN utility control statements. PGM_COBOL=YES affects the following function control statements:
• DECODE PSB
• DECODE ACB
• DECODE ACBPSB
• LIST PSB
• LIST ACB
In the PSB XREF BY TYPE - PSB Name Order report, COBOL is printed in the PSB LANGUAGE field.

**NO**
Prints LANG=ASSEM in the IMS PSBGEN utility control statements. PGM_COBOL=NO is the default value.

In the PSB XREF BY TYPE - PSB Name Order report, ASSEMBLE/COBOL is printed in the PSB LANGUAGE field.

**DECOPT** (DO)
This keyword affects all subsequent DECODE statements. The following options can be specified:

**CHECK_LEN=**
Identifies the DEDB DBDs and PSBs that were not generated by the IMS DBDGEN or IMS PSBGEN utility.

If a DBD or PSB was generated by a non-IMS macro, the length of the control block does not conform to the standard IMS control block length. The CHECK_LEN option checks the control block length to identify such DEDB DBDs and PSBs. The CHECK_LEN option is effective only for DBD and PSB members, and is not effective for ACB members.

**YES** Checks the control block length of each member and identifies the DEDB DBDs and PSBs that were not generated by the IMS DBDGEN or IMS PSBGEN utility.

If the DBD/PSB/ACB Reversal utility finds members that were generated by a non-IMS macro, the utility decodes only the portions of the DBD or PSB that conform to the standard DBD or PSB format and sets the return code to 4. The utility also prints FABN0084W messages in the SYSOUT Messages report to notify you about the identified members. If you also specify DECOPT COMMENT=YES, the FABN0084W messages are also printed on the comment line in the decoded DBD or PSB source code.

**NO** Does not check the control block length of each member. CHECK_LEN=NO is the default value.

**COMMENT=** (C=)
Specifies whether this program prints the comment lines (the heading part of the DATASET, SEGM, or PCB statement) from the decoded DBD or PSB sources.

**YES** The comment lines are printed. COMMENT=YES is the default value.

**NO** The comment lines are not printed.

**COMPRESS=** (COMP=)
Specifies whether the decoded DBD or PSB sources are printed in compressed format.

**Note:** In the DBD/PSB/ACB Reversal utility of Library Management Utilities and other prior products, non-compressed format was used.

**YES** The decoded sources are printed in compressed format. COMPRESS=YES is the default value.

**NO** The decoded sources are printed in noncompressed format.
PCB_LABEL=
Specifies whether to print the PCB name in the PCB label or on the PCBNAME control statement.

YES Prints the PCB name in the PCB label.
NO Prints the PCB name on the PCBNAME control statement. PCB_LABEL=NO is the default value.

SENSEG_PROCOPT=
Specifies to print the SENSEG PROCOPT value even when the value is the same as the PCB PROCOPT value.

YES Prints the value.
NO Does not print the value if the SENSEG PROCOPT value is the same as the PCB PROCOPT value. SENSEG_PROCOPT=NO is the default value.

VERSION_GENDATE=
Specifies to write the DBDGEN date and time on the VERSION control statement in the decoded DBD source.

During DBDGEN, the user can provide a character string on the VERSION control statement. If a character string is not provided, the DBDGEN utility automatically adds a 13-character time stamp (GENDATE time stamp), which represents the date and time when the DBDGEN completed. If you specify the VERSION_GENDATE=YES option on the DBD/PSB/ACB Reversal utility SYSIN control statement, the utility writes the GENDATE time stamp on the VERSION control statement in the decoded DBD source. This option is effective for DBD library members, but is not effective for ACB library members.

YES Prints the GENDATE time stamp on the VERSION control statement in the decoded DBD source. The format is MM/DD/YYHH.MM.
NO Does not print the GENDATE time stamp on the VERSION control statement, but prints it as a comment beside the VERSION control statement. VERSION_GENDATE=NO is the default value.

If you specify VERSION_GENDATE=YES for the DBD/PSB/ACB Reversal utility and use the generated DBD source as the input for the DBDGEN utility, the DBDGEN utility treats the GENDATE time stamp not as a time stamp but as a text. Consequently, the DBDGEN utility does not update the GENDATE time stamp with the latest time stamp. If you want the DBDs to always hold the latest DBDGEN time stamp, do not specify VERSION_GENDATE=YES.

In the following example, the options specified on the DECOPT statement affect two DECODE statements:

DECOPT COMMENT=NO,COMPRESS=NO
DECODE DBD ALL
DECODE PSB ALL
Function control statements

Function control statements specify the functions to be performed. A function control statement consists of a function keyword, a function keyword operand, and a function keyword option. The following function keywords can be used (the abbreviations that can be used are shown in parentheses):

DECODE (D)
This keyword specifies that this program re-creates the control statements of the IMS DBDGEN/PSBGEN utility in the SYSPUNCH data set or the DBDSRC/PSBSRC data set, or both.

LIST (L)
This keyword specifies that this program generates the following reports in the SYSPRINT data set:
- DBD XREF by Access reports
- PSB XREF by Type reports
- ACB(DBD) XREF by Access reports
- ACB(PSB) XREF by Type reports

DDNAMES (DDN)
This keyword specifies that this program generates the following reports, which contain information about cross-reference between DBDs and DDNAMES, in the SYSPRINT data set:
- DBD XREF by DDNAME report
- ACB(DBD) XREF by DDNAME report

PROCOPT (P)
This keyword specifies that this program generates the following reports, which contain information about processing options (PROCOPT) defined in PSBs, in the SYSPRINT data set:
- PCB PROCOPT report
- PCB/ACB(PSB) PROCOPT report

XREF (X)
This keyword specifies that this program generates the following reports, which contain information about cross-reference between DBDs or between DBDs and PSBs, in the SYSPRINT data set:
- DBD to DBD XREF report
- PSB to DBD XREF report
- ACB(DBD) to ACB(DBD) XREF report
- ACB(PSB) to ACB(DBD) XREF report

Note: The DBDNAME option can be specified only for XREF PSB and XREF ACB. For details, see “Function keyword operands” on page 295.

SEGREF (S)
This keyword with the SEGMENT option specifies that this program is to produce either of the following in the SYSPRINT data set.
- DBD Segment Reference report
- PSB Segment Reference report

POPTREF
This keyword specifies the criteria for selecting PSBs. Information about the PSBs that match the criteria is written to the PSB PROCOPT Reference report or the ACB PROCOPT Reference report in the SYSPRINT data set.
Criteria are defined by the SEARCHDBD and SEARCHOPT options. The
POPTREF keyword must be specified with those two options. The
SEARCHDBD option defines the DBD name criteria that are used to
identify the referenced DBDs. The SEARCHOPT option defines the
processing option (PROCOPT) criteria that are used to identify PSBs. Only
the PSBs that match the PROCOPT criteria and that refer to the DBDs that
are identified by the DBD name criteria are written to the report.

**UNREF**

This keyword specifies to generate the Unreferenced ACB(DBD) report in
the SYSPRINT data set. The only supported operand for this keyword is
ACB. No function keyword options are supported for this keyword.

**LISTLIB**

This keyword specifies to generate the DBD or PSB library member list
report in the SYSPRINT data set. The report includes the following
information about the members in the data sets that are concatenated to
DBDLIB DD or PSBLIB DD:

- IMS version that generated the DBD or PSB member
- Generation date and time
- Size of the member record

**Function keyword operands**

The following operands can be specified on the function control statements (the
abbreviation that can be used is shown in parentheses):

**DBD (D)**

This operand specifies that the operation is to be performed on one or
more DBDs.

**PSB (P)**

This operand specifies that the operation is to be performed on one or
more PSBs.

**ACB (A)**

This operand specifies that the operation is to be performed on one or
more PSB-type ACBs and on the DBD-type ACBs that those PSB-type
ACBs refer to.

**ACBPSB (AP)**

This operand specifies that the operation is to be performed on one or
more PSB-type ACBs. This operand is valid only for the DECODE
keyword.

**ACBDBD (AD)**

This operand specifies that the operation is to be performed on one or
more DBD-type ACBs. This operand is valid only for the DECODE
keyword.

**Function keyword options**

The following options can be specified on the function control statements (the
abbreviation that can be used is shown in parentheses):

**ALL (A)**

This option specifies that the operation is to be performed on all DBD,
PSB, or ACB members of the library depending upon the operand specified
in the control record. Specifying ALL (A) causes the DBD/PSB/ACB
Reversal to process all the members in the data sets that are concatenated to DBDLIB DD, PSBLIB DD, or ACBLIB DD.

If two or more data sets in the concatenation contain a member with the same name, DBD/PSB/ACB Reversal processes only the first one in the concatenation.

If the ACB operand is specified, the ACB reversal function processes all PSB-type ACBs and the DBD-type ACBs that those PSB-type ACBs refer to.

If the ACBPSB operand is specified, the ACB reversal function processes all PSB-type ACBs.

If the ACBDBD operand is specified, the ACB reversal function processes all DBD-type ACBs.

**INCLUDE= (I=)**

This option specifies that the operation is to be performed on the DBD, PSB, or ACB members specified after the equal sign (depending on the operand specified in the control record).

**Note:** You can specify a wildcard in any position of a character string. The asterisk (*) and the percent sign (%) are supported as wildcard characters. An asterisk represents 0 - 8 characters, and the percent sign (%) represents a single character. If two or more asterisks are specified sequentially, only the first asterisk is recognized.

When the ACB operand is specified, the specified ACBs must be of PSB type. The ACB reversal function processes the specified PSB-type ACBs and the DBD-type ACB members that those PSB-type ACBs refer to. If the specified ACB is not PSB type, an error message is issued and this member is skipped.

When the ACBPSB operand is specified, the specified ACBs must be of PSB type. The ACB reversal function processes the specified PSB-type ACBs. If the specified ACB is not of PSB type, an error message is issued and this member is skipped.

When the ACBDBD operand is specified, the specified ACBs must be of DBD type. The ACB reversal function processes the specified DBD-type ACBs. If the specified ACB is not of DBD type, an error message is issued and this member is skipped.

**EXCLUDE= (E=)**

This option specifies that the operation is to be performed on all DBD, PSB, or ACB members other than those specified after the equal sign (depending on the operand specified in the control record). If concatenated data sets are specified for the DBDLIB DD, PSBLIB DD, or ACBLIB DD statement, the DBD/PSB/ACB Reversal processes only the first data set.

You can use wildcards for multiple character replacement. The method of using them is the same as for the INCLUDE (I) option.

If the ACB operand is specified, the specified ACBs must be of PSB type. The ACB reversal function processes the PSB-type ACBs that are not specified and the DBD-type ACBs that those PSB-type ACBs refer to.

If the ACBPSB operand is specified, the specified ACBs must be of PSB type. The ACB reversal function processes the PSB-type ACBs that are not specified.
If the ACBDBD operand is specified, the specified ACBs must be of DBD type. The ACB reversal function processes the DBD-type ACBs that are not specified.

**SEGMENT= (S=)**

This option specifies that the operation is to be performed on the segments specified after the equal sign. This option is valid only with the keyword SEGREF; if it is specified without the keyword SEGREF, it is ignored. You can specify only one segment name for the SEGMENT= option. However, a wildcard character can be specified as the segment name. Its use is the same as that of the INCLUDE (I) option.

**DBDNAME= (D=)**

This option is valid only in the XREF PSB control statement or the XREF ACB control statement. The PSB to DBD XREF report or the ACB(PSB) to ACB(DBD) XREF report has two parts: the reference report and the referenced report. If the DBDNAME= option is specified, only the referenced report part is printed for the specified DBDs. The DBD name can be specified with its exact name, or with the use of wildcards. If this option is accepted, message FABN0063I is issued.

**PCBNAMEX= (P=)**

This option specifies the prefix of the PCB names within 1 - 4 characters. This option is valid only with the keyword DECODE and operands PSB, ACB, or ACBPSB. The following is an example of the control card:

```
DECODE PSB INCLUDE=psbname PCBNAMEX=prfx
```

If this option is specified and there is one or more PCBs that are not named in the PSB, DBD/PSB/ACB Reversal assigns the PCB names on the PCBNAME parameters of PCB statements as PCBNAME=prfxnnnn, where prfx is the specified prefix and nnnn is the PCB number.

**SEARCHDBD=**

This option specifies the DBD name criteria to use to filter the referenced DBDs. This option is a required option for the POPTRF keyword. If this option is specified without the POPTRF keyword, it is ignored.

You can specify the name of a DBD member or use the asterisk (*) wildcard. An asterisk represents 0 - 8 characters. If two or more asterisks are specified sequentially, only the first asterisk is recognized.

This option can be specified on a new line.

**SEARCHOPT=**

This option specifies the processing option (PROCOPT) criteria, in 1 - 4 characters, to use to filter the PSBs. This option is a required option for the POPTRF keyword. The option must follow the SEARCHDBD option. If this option is specified without the POPTRF keyword, it is ignored.

You can specify the PROCOPT criteria by using the letters that correspond to the PROCOPT values and by using an asterisk (*) or percent sign (%) as a wildcard character. An asterisk represents 0 - 8 characters, and the percent sign represents a single character. If two or more asterisks are specified sequentially, only the first asterisk is recognized. An asterisk and a percent sign cannot be specified together.

You can specify the letters in any order because the order of the letters in the SEARCHOPT option is not considered. However, if you use both letters and wildcard characters, the letters must precede the wildcard characters.

This option can be specified on a new line.
The following table contains examples for specifying the PROCOPT criteria. In these examples, the following PROCOPT values are defined in the PSBs: G, GO, GP, GOP, GON, GHT, A, AP, and PA.

<table>
<thead>
<tr>
<th>PROCOPT criteria</th>
<th>Matched PROCOPT values</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEARCHOPT=G</td>
<td>G</td>
</tr>
<tr>
<td>SEARCHOPT=GO</td>
<td>GO</td>
</tr>
<tr>
<td>SEARCHOPT=AP</td>
<td>AP, PA</td>
</tr>
<tr>
<td>SEARCHOPT=*</td>
<td>Any PROCOPT value</td>
</tr>
<tr>
<td>SEARCHOPT=G*</td>
<td>G, GO, GP, GOP, GON, GHT</td>
</tr>
<tr>
<td>SEARCHOPT=P*</td>
<td>GP, GOP, AP, PA</td>
</tr>
<tr>
<td>SEARCHOPT=%</td>
<td>G, A</td>
</tr>
<tr>
<td>SEARCHOPT=G%</td>
<td>GO, GP</td>
</tr>
<tr>
<td>SEARCHOPT=P%</td>
<td>GP, GO, AP, PA</td>
</tr>
<tr>
<td>SEARCHOPT=G%%</td>
<td>GOP, GON, GHT</td>
</tr>
</tbody>
</table>

Quick reference for control statements and DD statements

The following table lists the DBD/PSB/ACB Reversal functions, control statements, and DD statements.

Table 21. DBD/PSB/ACB Reversal functions, control statements, and DD statements

<table>
<thead>
<tr>
<th>Function</th>
<th>Control key</th>
<th>Control operand</th>
<th>Required and optional DD statements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>SYS PRINT</td>
</tr>
<tr>
<td>DBD reversal</td>
<td>DECODE</td>
<td>DBD</td>
<td>R</td>
</tr>
<tr>
<td>PSB reversal</td>
<td>DECODE</td>
<td>PSB</td>
<td>R</td>
</tr>
<tr>
<td>ACB reversal</td>
<td>DECODE</td>
<td>ACB</td>
<td>R</td>
</tr>
<tr>
<td>ACB reversal (PSB-type only)</td>
<td>DECODE</td>
<td>ACBPSB</td>
<td>R</td>
</tr>
<tr>
<td>ACB reversal (DBD-type only)</td>
<td>DECODE</td>
<td>ACBDBD</td>
<td>R</td>
</tr>
<tr>
<td>DBD summary</td>
<td>LIST</td>
<td>DBD</td>
<td>R</td>
</tr>
<tr>
<td>PSB summary</td>
<td>LIST</td>
<td>PSB</td>
<td>R</td>
</tr>
<tr>
<td>ACB summary</td>
<td>LIST</td>
<td>ACB</td>
<td>R</td>
</tr>
<tr>
<td>PCB PROCOPT</td>
<td>PROCOPT</td>
<td>PSB</td>
<td>R</td>
</tr>
<tr>
<td>PCB/ACB (PSB) PROCOPT</td>
<td>PROCOPT</td>
<td>ACB</td>
<td>R</td>
</tr>
<tr>
<td>DBD-DBD XREF</td>
<td>XREF</td>
<td>DBD</td>
<td>R</td>
</tr>
<tr>
<td>PSB-DBD XREF</td>
<td>XREF</td>
<td>PSB</td>
<td>R</td>
</tr>
<tr>
<td>ACB(DBD)-ACB(DBD) XREF</td>
<td>XREF</td>
<td>ACB</td>
<td>R</td>
</tr>
<tr>
<td>ACB(DBD)-ACB(DBD) XREF</td>
<td>XREF</td>
<td>ACB</td>
<td>R</td>
</tr>
<tr>
<td>DDname XREF</td>
<td>DDNAMES</td>
<td>DBD</td>
<td>R</td>
</tr>
<tr>
<td>DDname XREF (ACB(DBD))</td>
<td>DDNAMES</td>
<td>ACB</td>
<td>R</td>
</tr>
<tr>
<td>Function</td>
<td>Control keyword</td>
<td>Control operand</td>
<td>Required and optional DD statements (O: Optional R: Required)</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SYS PRINT</td>
</tr>
<tr>
<td>DBD SEGMENT reference</td>
<td>SEGREF</td>
<td>DBD</td>
<td>R</td>
</tr>
<tr>
<td>PSB SEGMENT reference</td>
<td>SEGREF</td>
<td>PSB</td>
<td>R</td>
</tr>
<tr>
<td>PSB PROCOPT reference</td>
<td>POPTREF</td>
<td>PSB</td>
<td>R</td>
</tr>
<tr>
<td>ACB(PSB) PROCOPT reference</td>
<td>POPTREF</td>
<td>ACB</td>
<td>R</td>
</tr>
<tr>
<td>UN-REFERENCED ACB(DBD)</td>
<td>UNREF</td>
<td>ACB</td>
<td>R</td>
</tr>
<tr>
<td>DBD library member list</td>
<td>LISTLIB</td>
<td>DBD</td>
<td>R</td>
</tr>
<tr>
<td>PSB library member list</td>
<td>LISTLIB</td>
<td>PSB</td>
<td>R</td>
</tr>
</tbody>
</table>

**Notes:**
1. When the DBD reversal function or the ACB reversal function (DBD-type only) is run, either the SYSPUNCH data set or the DBDSRC data set must be specified.
2. When the PSB reversal function or the ACB reversal function (PSB-type only) is run, either the SYSPUNCH data set or the PSBSRC data set must be specified.
3. When the ACB reversal function is run, either the SYSPUNCH data set or the DBDSRC and PSBSRC data sets (both) must be specified.
4. When the DBD/PSB/ACB reversal function or the DBD/PSB Segment Reference report function is run, the mapper input generate function is also executed, and input control statements are written to the MAPOUT DD.
5. When the ACB reversal or the report function is run with OPTION ACB_GSAM=YES, the DD statement is required.
JCL examples for the DBD/PSB/ACB Reversal utility

The following topics provide JCL examples for running the DBD/PSB/ACB Reversal utility to re-create IMS DBDGEN and PSBGEN control statements.

Example: Re-creating the sources from DBDs and PSBs

The following figure shows example JCL for running a job that re-creates the IMS DBDGEN and PSBGEN control statements from the DBD and PSB libraries and adds members that contain each source to the partitioned data sets that are specified by the DBDSRC DD statement and the PSBSRC DD statement.

The first control statement specifies to re-create the sources for all DBDs in the DBD library. The second control statement specifies to re-create the sources for all PSBs in the PSB library.

All sources are printed in the SYSPUNCH data set.

```
//REVERS EXEC PGM=FABNRVRS
//STEPLIB DD DSN=HPS.SHPMLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPUNCH DD SYSOUT=A
//DBDSRC DD DSN=PDS.DBDSRC,DISP=SHR
//PSBSRC DD DSN=PDS.PSBSRC,DISP=SHR
//SYSIN DD *
    DECODE DBD ALL
    DECODE PSB ALL
/*
```

Figure 102. Example of re-creating the sources from DBDs and PSBs
Example: Re-creating the sources from ACBs

The following figure shows example JCL for running a job that re-creates the IMS DBDGEN and PSBGEN control statements from the specified ACB library and adds members that contain each source to the partitioned data sets that are specified by the DBDSRC DD statement and the PSBSRC DD statement.

The first control statement specifies to re-create the sources for all PSB-type ACBs and the DBD-type ACBs that those PSB-type ACBs refer to. The second control statement specifies to re-create a source for the named DBD-type ACB. The third control statement specifies to re-create a source for the named PSB-type ACB.

All sources are printed in the SYSPUNCH data set.

//REVERS EXEC PGM=FABNRVRS,REGION=0M
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPUNCH DD SYSOUT=A
//DBDSRC DD DSN=PDS.DBDSRC,DISP=SHR
//PSBSRC DD DSN=PDS.PSBSRC,DISP=SHR
//SYSSN DD *

  DECODE ACB ALL
  DECODE ACBDDB INCLUDE=HDAM01
  DECODE ACBPSB INCLUDE=PSB01A

/*

Figure 103. Example of re-creating the sources from ACBs
Example: Obtaining DBD library information

The following figure shows example JCL for running a job in which DBD library summary information is obtained by running three DBD/PSB/ACB Reversal utility functions.

The following information is obtained:

- DBD member access method information for each DBD member
- DBD member cross-reference
- DBD member and ddname cross-reference

```plaintext
//REVERS EXEC PGM=FABNRVRS
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
   LIST DBD ALL
   XREF DBD ALL
   DDNAMES DBD ALL
/*
```

Figure 104. Example of obtaining DBD library information

Example: Obtaining PSB library information

The following figure shows example JCL for running a job in which PSB library summary information is obtained by running three DBD/PSB/ACB Reversal utility functions.

The following information is obtained:

- PSB-type and language information for each PSB member
- PCB-related information for each PSB (PCB-type, LTERM name referred to for TP PCB, or DBD name referred to for DB PCB and GSAM PCB)
- PSB member and DBD cross-reference

```plaintext
//REVERS EXEC PGM=FABNRVRS
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
   LIST PSB ALL
   PROCOPT PSB ALL
   XREF PSB ALL
/*
```

Figure 105. Example of obtaining PSB library information
Example: Obtaining control statement source and Mapper input

The following figure shows example JCL for a procedure in which IMS DBDGEN and PSBGEN utility control statements are obtained and the DBD/PSB/ACB Mapper program is run.

The first step runs the reversal function with the MAPOUT data set specified, and the second step runs DBD/PSB/ACB Mapper. You must supply the DECODE statement with the target member name specified as the SYSIN control statement to the first step.

DBD/PSB/ACB Reversal outputs a list of member names to the MAPOUT data set. The names are used as SYSIN control statements to DBD/PSB/ACB Mapper.

By using DBD/PSB/ACB Reversal and DBD/PSB/ACB Mapper together, you can obtain the DBDGEN/PSBGEN source statements as well as a visual representation of the control blocks.

---

Figure 106. Example of input generation functions of Reversal and Mapper

---

Chapter 8. DBD/PSB/ACB Reversal utility  303
Output from the DBD/PSB/ACB Reversal utility

Output from DBD/PSB/ACB Reversal consists of the SYSOUT data set, the SYSPUNCH data set, the DBDSRC data set, the PSBSRC data set, the SYSPRINT data set, the MAPOUT data set, and the OPTPR data set.

SYSOUT data set

The SYSOUT data set contains activity log and error messages for all of the functions of the DBD/PSB/ACB Reversal program.

The following figure shows messages that are generated in the SYSOUT data set.

Figure 107. Messages in the SYSOUT data set
SYSPUNCH data set

The SYSPUNCH data set contains the IMS DBDGEN or PSBGEN utility control statements if the SYSIN data set contains one or more DECODE DBD, DECODE PSB, DECODE ACB, DECODE ACBPSB, or DECODE ACBDBD control statements.

Subsections:

- "Re-created utility control statements"
- "Restrictions on the generated control statements" on page 306

Re-created utility control statements

The following figures show examples of the IMS DBDGEN and PSBGEN utility control statements created by DBD/PSB/ACB Reversal.

For the control statements not specified in the source, DBD/PSB/ACB Reversal always decodes the default value defined explicitly by IMS DBDGEN or PSBGEN utility.

Note: If you want to eliminate the comment lines, which are the heading of the DATASET, SEGM, or PCB statement, specify DECOPT COMMENT=NO in the first control statement of the SYSIN data set.

```
TITLE 'ASSEMBLE OF DBDNAME=DSCLSDVN'
* DSNAME=IMSVS.DBDLIB
* VOL=IMSVS
* DBDGEN DATE 11/22/2015 TIME 19.49
* DECODE DATE 12/01/2015 TIME 10.18.54
* IMS VERSION 13.1

DBD NAME=DSCLSDVN,ACCESS=(HDAM,VSAM),
RMNAME=(DFSHDC40,1,2,6),PASSWD=YES,
VERSION= DATE 11/22/2015 TIME 19.49
******************************************************************************
* DATASET GROUP NUMBER 1
******************************************************************************
DSG001 DATASET DD1=DSCLSDV0,SIZE=(2048),SCAN=3
******************************************************************************
* SEGMENT NUMBER 1
******************************************************************************
SEGMM NAME=SSCLSP00,PARENT=0,BYTES=(100,5),RULES=(LVV,LAST),
PTR=(TWIN,,,,)
FIELD NAME=(ECLASSR,SEQ,U),START=3,BYTES=3,TYPE=C
FIELD NAME=(ECLSP0LE),START=1,BYTES=2,TYPE=C
LCHILD NAME=(SSCRSP23,DSCRSDVN),PTR=SNGL,PAIR=SSCLSV11,
RULES=HERE
******************************************************************************
* SEGMENT NUMBER 2
******************************************************************************
SEGMM NAME=SSCLSV11,PARENT=((SSCLSP00,)),PTR=PAIRED,
SOURCE=((SSCRSP23,DSCRSDVN)),
FIELD NAME=(ECLASSNR,SEQ,U),START=21,BYTES=6,TYPE=C
DBDGEN FINISH END
```

Figure 108. IMS DBDGEN utility control statements re-created by DBD/PSB/ACB Reversal
Restrictions on the generated control statements

When the IMS DBDGEN utility processes the control statements issued by the DBD/PSB/ACB Reversal utility, the following restrictions apply:

- The order of the FIELD, LCHILD, and XDFLD statements that follow the SEGM statement is not the same as the user-required order in the DBD control statements. DBD/PSB/ACB Reversal generates all of the FIELD statements that belong to the segment following the SEGM statements, and then produces, if they exist, the LCHILD statements with paired XDFLD statements. This does not affect the database being accessed.

- If the VERSION parameter on the DBD statement has a time stamp value, DBD/PSB/ACB Reversal decodes the time stamp value as an Assembler comment statement.

When the IMS PSBGEN utility processes the control statements issued by the DBD/PSB/ACB Reversal, the following restrictions apply:

- The PCB label and the PCBNAME parameter in the PCB statement are mutually exclusive. Whether a PCB label or a PCBNAME parameter is used to decode a PCB name depends on the version of IMS that generated the PSB:
  - For PSBs that were generated by IMS Version 3 or higher, if the PCB_LABEL=YES option (uses the PCB label) is not specified, DBD/PSB/ACB Reversal uses the PCBNAME parameter to decode the PCB name.
  - For PSBs that were generated by IMS Version 2 Release 2, DBD/PSB/ACB Reversal uses the PCB label to decode the PCB name.

- If the PGM_COBOL=YES option (prints LANG=COBOL) is not specified, DBD/PSB/ACB Reversal always decodes the PSBGEN statement as PSBGEN LANG=ASSEM, even if the statement is defined as PSBGEN LANG=COBOL or PSBGEN LANG=, because there is no difference between the PSBs.
• DBD/PSB/ACB Reversal always decodes the TP PCB statement as PCB TYPE=TP,TERM=nnnn, even if it is defined as PCB TYPE=TP,NAME=nnnn, because there is no difference between the two PSBs.

When the IMS DBDGEN utility processes the control statements issued by the ACB reversal function of DBD/PSB/ACB Reversal, the following restrictions apply:
• When DBD/PSB/ACB Reversal cannot obtain complete segment name information from one or more ACBs, the program decodes the segment name as follows:
  NAME=$FABNnnn (nnn is the segment code)
Then the program writes FABN0039W message to both SYSOUT and SYSPUNCH data set.
• Because the ACB library contains no information on the SIZE parameter, the second RECORD parameter, or the DEVICE parameter of the DATASET statement, the DBD/PSB/ACB Reversal cannot decode these parameters.
• Because the ACB library contains no information on the FREQ parameter of the SEGM statement, the DBD/PSB/ACB Reversal cannot decode the FREQ parameter.
• Because DBD/PSB/ACB Reversal cannot obtain paired segment name information on the LCHILD statement for virtually paired logical relationship, the program cannot decode the PAIR parameter.
• Because DBD/PSB/ACB Reversal cannot obtain segment name information and database name information of the SOURCE parameter for virtually paired logical relationship, the program decodes these names as follows:
  SOURCE=(($SEGMnnn,DATA,$DBDnmm)) (nnn is a sequential number in a DBD)
Then the program writes message FABN0065W to both the SYSOUT and the SYSPUNCH data set.
• Because the IMS ACBGEN does not generate any DBD-type ACB for logical DBD, DBD/PSB/ACB Reversal decodes the logical DBD by using information from the PSB-type ACB that refers to the logical DBD when specifying the PSB-type ACB with the ACB keyword. Therefore the SEGM statements are decoded only for the sensitive segments, and the order of the SEGM statements might be different from the user-required order.
• Because the ACB library contains no information on the GSAM database, DBD/PSB/ACB Reversal does not decode the DBD control statements for the GSAM database if ACB_GSAM=YES is not specified for the runtime option.
• When decoding a DBD-type ACB for a DEDB, a PSB-type ACB, which references the DBD, is also used to for obtaining the DBD VERSION= parameter value. If a problem occurs when reading the PSB-type ACB, the DBD VERSION= parameter value is not decoded.
• If the index target segment type is assumed to be the index source segment, DBD/PSB/ACB Reversal prints the XDFLD SEGMENT parameter without a value.

When IMS PSBGEN utility processes the control statements issued by the ACB reversal function of the DBD/PSB/ACB Reversal, the following restrictions apply:
• In a DEDB database, DBD/PSB/ACB Reversal always decodes the POS parameter of the PCB statement as POS=S.
• DBD/PSB/ACB Reversal decodes the INDICES parameter of the SENSEG statement into the following format:
  INDICES=(($DBD0001,$DBD0002)
DBD/PSB/ACB Reversal provides the specified number of index DBDs, but it does not get the real DBD name from a PSB-type ACB. Therefore it assigns an alternative DBD name that contains a four-digit sequential number in a PSB. The program then writes message FABN0055W to the SYSOUT data set. Replace each DBD name that is assigned with the real index DBD name manually, to regenerate DBD/PSB and to build ACB from the sources generated by DBD/PSB/ACB Reversal.

- DBD/PSB/ACB Reversal always decodes the REPLACE parameter of the SENFLD statement as REPLACE=YES.
- DBD/PSB/ACB Reversal program does not decode the GSAM PCB in the ACB library if ACB_GSAM=YES is not specified for the runtime option.

**DBDSRC data set**

The DBD/PSB/ACB Reversal program creates the DBDGEN utility control statement in the PDS or PDSE specified in the DBDSRC DD statement. The DBDGEN utility control statement is the same as the one created in the SYSPUNCH data set.

If the specified data set is not a PDS nor a PDSE when the DBD or ACB reversal function is run, the program ends abnormally with an open error.

The following restrictions apply when a PDS or a PDSE is specified in the DBDSRC DD statement:

- If the member name specified exists in the PDS or PDSE specified, the program replaces the specified member name with a new member and the ALIAS member name.
- If the ALIAS member name is specified as a decoded member name in the DBDLIB library or the ACBLIB library, the program decodes the member with the real member name, not the ALIAS member name.

**PSBSRC data set**

The DBD/PSB/ACB Reversal program creates the PSBGEN utility control statement in the PDS or PDSE specified in the PSBSRC DD statement. The PSBGEN utility control statement is the same as the one created in the SYSPUNCH data set.

If the specified data set is not a PDS nor a PDSE when the PSB or ACB reversal function is run, the program ends abnormally with an open error.

The following restrictions apply when a PDS or a PDSE is specified in the PSBSRC DD statement:

- If the member name specified exists in the PDS or PDSE specified, the program replaces the specified member name with a new member and the ALIAS member name.
- If the ALIAS member name is specified as a decoded member name in the PSBLIB library or the ACBLIB library, the program decodes the member with the real member name, not the ALIAS member name.

**SYSPRINT data set**

The SYSPRINT data set contains the reports corresponding to the specified control statements in the SYSIN data set.
XREF by Access reports for DBD and ACB(DBD)

The XREF by Access reports for DBDs and DBD-type ACBs contain information about the specified DBDs or DBD-type ACBs and the DL/I access method defined in each DBD or DBD-type ACB.

If a LIST DBD control statement is specified, the following two cross-reference reports are generated:
- DBD XREF by Access - DBD Name Order report
- DBD XREF by Access - Access Order report

If a LIST ACB control statement is specified, the following two cross-reference reports are generated:
- ACB(DBD) XREF by Access - ACB(DBD) Name Order report
- ACB(DBD) XREF by Access - Access Order report

Sample report: DBD Name Order report

This report lists the specified DBD names and the DL/I access method defined in each DBD. It is ordered alphabetically by DBD name. For ALIAS-type members, the actual name is shown in parentheses. If the ALIAS-type member is specified by the INCLUDE option, the actual name is not shown.

The following figure shows an example of the DBD XREF by Access - DBD Name Order report.

<table>
<thead>
<tr>
<th>DBDNAME</th>
<th>ACCESS METHOD</th>
<th>DBDNAME</th>
<th>ACCESS METHOD</th>
<th>DBDNAME</th>
<th>ACCESS METHOD</th>
<th>DBDNAME</th>
<th>ACCESS METHOD</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDAM01</td>
<td>HDAM,VSAM</td>
<td>HIDAM01</td>
<td>HIDAM,OSAM</td>
<td>INDEXDB</td>
<td>INDEX,VSAM,SGL</td>
<td>PHDAM01</td>
<td>PHIDAM,OSAM</td>
</tr>
<tr>
<td>HIDAM04</td>
<td>PHIDAM,VSAM</td>
<td>PHIDAM01</td>
<td>PHIDAM,OSAM</td>
<td>PHIDAM04</td>
<td>PHIDAM,VSAM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSINDEX01</td>
<td>PSINDEX,VSAM</td>
<td>SINDEX01</td>
<td>INDEX,VSAM,SGL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 110. DBD XREF by Access - DBD Name Order report

Sample report: Access Order report

This report lists the DL/I access methods, the total number of the DBDs using each access method, and the DBD names.

The following figure shows an example of the DBD XREF by Access - Access Order report.
The XREF by Type reports for PSBs and PSB-type ACBs contain information about the specified PSBs or PSB-type ACBs, the type, and the language information for each PSB or PSB-type ACB.

If a LIST PSB control statement is specified, the following two cross-reference reports are generated:

- PSB XREF by Type - PSB Name Order report
- PSB XREF by Type - Type Order report

If a LIST ACB control statement is specified, the following two cross-reference reports are generated:

- ACB(PSB) XREF by Type - ACB(PSB) Name Order report
- ACB(PSB) XREF by Type - Type Order report

**Sample report: PSB Name Order report**

This report lists the specified PSB names with the PSB type and language information about each PSB. For ALIAS-type members, the actual name is shown in parentheses. If the ALIAS-type member is specified by the INCLUDE option, the actual name is not shown.

**Tip:** The PSB LANGUAGE column indicates the language (LANG= value) in the decoded IMS PSBGEN utility control statement. You can use the PGM_COBOL option to change the LANG= value.

The following figure shows an example of the PSB XREF by Type - PSB Name Order report.
Sample report: Type Order report

This report lists the PSB types and the total number of the PSBs for each type, and the PSB names.

The following figure shows an example of the PSB XREF by Type - Type Order report.

XREF by DDname reports for DBD and ACB(DBD)

The XREF by DDname reports for DBDs and DBD-type ACBs contain information about the DBDs and the ddnames that each DBD or DBD-type ACB refers to, and ddnames and the DBDs or DBD-type ACBs that refer to each ddname.
If a DDNAMES DBD statement is specified, the following two XREF by DDname reports are generated:

- DBD XREF by DDname - reference report
- DBD XREF by DDname - referenced report

If a DDNAMES ACB statement is specified, the following two XREF by DDname reports are generated:

- ACB(DBD) XREF by DDname - reference report
- ACB(DBD) XREF by DDname - referenced report

Sample report: Reference report

This report lists the DBDs and the ddnames that each DBD refers to.

The following figure shows an example of the DBD XREF by DDname - reference report.

Figure 114. DBD XREF by DDname - reference report
Sample report: Referenced report

This report lists the ddnames and the DBDs that refer to each ddname.

The following figure shows an example of the DBD XREF by DDname - referenced report.

PCB PROCOPT reports for PSB and ACB(PSB)

If a PROCOPT PSB control statement is specified, a PCB PROCOPT report, which contains information about the PCB processing options (PROCOPT) in the PSB, is generated. If a PROCOPT ACB control statement is specified, a PCB/ACB(PSB) PROCOPT report, which contains information about the PCB processing options (PROCOPT) in the PSB-type ACB, is generated.
Sample report

The following figure shows an example of the PCB PROCOPT report.

![Table of PCB PROCOPT report](image)

**Report field descriptions**

The meaning of each column is as follows:

**PCB NO**

This field shows the sequential number assigned to the PCB in the PSB.

**TYPE**

This field shows the PCB type.
**REF DBD/LTERM**

If the TYPE is DB PCB or GSAM PCB, this field shows the DBD name specified in the PCB. If the TYPE is TP PCB, this field shows the logical terminal names specified in the PCB.

**PROCSEQ**

This field shows the Processing Options (PROCSEQ) defined in each PCB.

**PROCSEQ(D)**

This field shows the secondary index DBD name that was specified in the PCB PROCSEQ= or PROCSEQD= statement of the PSBGEN utility.

**XREF reports for DBD to DBD and ACB(DBD) to ACB(DBD)**

The XREF reports for DBD to DBD or DBD-type ACB to DBD-type ACB contain cross-reference information between DBDs or DBD-type ACBs.

If an XREF DBD statement is specified, the following two XREF reports are generated:

- DBD to DBD XREF - reference report
- DBD to DBD XREF - referenced report

If an XREF ACB statement is specified, the following two XREF reports are generated:

- ACB(DBD) to ACB(DBD) XREF - reference report
- ACB(DBD) to ACB(DBD) XREF - referenced report

**Sample report: Reference report**

This report lists the DBDs and the other DBDs that each DBD refers to.

The following figure shows an example of the DBD to DBD XREF - reference report.
Sample report: Referenced report

This report lists the DBDs and the other DBDs that refer to each DBD.

The following figure shows an example of the DBD to DBD XREF - referenced report.
XREF reports for PSB to DBD and ACB(PSB) to ACB(DBD)

The XREF reports for PSB to DBD and PSB-type ACB to DBD-type ACB contain cross-reference information between DBDs and PSBs or PSB-type ACBs and DBD-type ACBs.

If an XREF PSB statement is specified, the following two XREF reports are generated:
- PSB to DBD XREF - reference report
- PSB to DBD XREF - referenced report

If an XREF ACB statement is specified, the following two XREF reports are generated:
- ACB(PSB) to ACB(DBD) XREF - reference report
- ACB(PSB) to ACB(DBD) XREF - referenced report

Tip: If you specify ACB_REFERENCED=NO and the utility finds a DBD-type ACB that is not referenced by any PSB-type ACBs, NONE is printed for the DBD-type ACB in the REFERENCED BY THE FOLLOWING PSBS column.

Sample report: Reference report

This report lists the PSBs and the DBDs that each PSB refers to.

The following figure shows an example of the PSB to DBD XREF - reference report.
Sample report: Referenced report

This report lists the DBDs and the PSBs that refer to each DBD.

The following figure shows an example of the PSB to DBD XREF - referenced report.
If a SEGREF DBD statement is specified, this report that summarizes the relationship between DBDs and the segment specified by the SEGMENT option is generated.

Subsections:
- “Sample report” on page 320
- “Report field descriptions” on page 320

Segment reference report for DBD

If a SEGREF DBD statement is specified, this report that summarizes the relationship between DBDs and the segment specified by the SEGMENT option is generated.

Subsections:
- “Sample report” on page 320
- “Report field descriptions” on page 320
**Sample report**

The following figure shows an example of the DBD Segment Reference report.

<table>
<thead>
<tr>
<th>ORIGINAL SEGMENT</th>
<th>ORIGINAL DBD</th>
<th>ACCESS STATEMENT</th>
<th>AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHIDAMD1</td>
<td>PHIDAM01</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
<tr>
<td>PHIDAMD2</td>
<td>PHIDAM01</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
<tr>
<td></td>
<td>PHIDAMD3</td>
<td></td>
<td>SEGMENT PARENT= PHYSICAL PARENT</td>
</tr>
<tr>
<td>PHIDAMD3</td>
<td>PHIDAM01</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
<tr>
<td></td>
<td>PHIDAMD4</td>
<td></td>
<td>SEGMENT PARENT= PHYSICAL PARENT</td>
</tr>
<tr>
<td>PHIDAMD4</td>
<td>PHIDAM01</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
<tr>
<td>PHIDAMRT</td>
<td>PHIDAM01</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
<tr>
<td></td>
<td>PHIDAMD1</td>
<td></td>
<td>SEGMENT PARENT= PHYSICAL PARENT</td>
</tr>
<tr>
<td>SECDEP01</td>
<td>SECONDO1</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
<tr>
<td>SECDEP02</td>
<td>SECONDO1</td>
<td></td>
<td>SEGMENT PARENT= PHYSICAL PARENT</td>
</tr>
<tr>
<td></td>
<td>SECINDEX</td>
<td></td>
<td>LCHILD NAME= INDEX TARGET</td>
</tr>
<tr>
<td>SECDEP02</td>
<td>SECONDO1</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
<tr>
<td></td>
<td>SECDEP01</td>
<td></td>
<td>XDFLD SEGMENT= INDEX SOURCE</td>
</tr>
<tr>
<td>SECDEP03</td>
<td>SECONDO1</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
<tr>
<td>SECINDEX</td>
<td>SECONDO1</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
<tr>
<td></td>
<td>SECDEP01</td>
<td></td>
<td>LCHILD NAME= INDEX POINTER</td>
</tr>
<tr>
<td>SECROOT</td>
<td>SECONDO1</td>
<td>PHIDAM</td>
<td>SEGMENT NAME= ORIGINAL</td>
</tr>
</tbody>
</table>

**Report field descriptions**

The meaning of each column is as follows:

**ORIGINAL SEGMENT**

This field shows the segment name that is specified by the SEGMENT option.

**ORIGINAL DBD**

This field shows the DBD name in which each ORIGINAL SEGMENT is defined by the SEGM NAME= statement of the DBDGEN utility.

**REFERENCE SEGMENT**

This field shows the segment name which is defined by the SEGM NAME= statement of the DBDGEN utility. It refers to each ORIGINAL SEGMENT in the succeeding DBDGEN control statement such as SEGMENT PARENT= or LCHILD NAME=.

**Note:** The asterisk (*) means that the name is the same name as the ORIGINAL SEGMENT.

**REFERENCE DBD**

This field shows the DBD name in which the REFERENCE SEGMENT is defined.
Note: The asterisk (*) means that the name is the same name as the ORIGINAL DBD.

**ACCESS**
This field shows the DL/I access method for ORIGINAL DBD or REFERENCE DBD.

**STATEMENT**
This field shows the DBDGEN utility control statement by which ORIGINAL SEGMENT is referred to.

**Segment reference report for PSB**
If a SEGREF PSB statement is specified, the PSB Segment Reference report is generated. This report summarizes the relationship between PSBs and the segment specified by the SEGMENT option.

Subsections:
- "Sample report"
- "Report field descriptions"

**Sample report**
The following figure shows an example of the PSB Segment Reference report.

```
<table>
<thead>
<tr>
<th>SEGMENT NAME</th>
<th>PCB</th>
<th>PCB SEGMENT</th>
<th>PSB NAME</th>
<th>NO</th>
<th>REF DBD</th>
<th>PROCOPT</th>
<th>PCB SEGMENT</th>
<th>PSB NAME</th>
<th>NO</th>
<th>REF DBD</th>
<th>PROCOPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>HYS000A</td>
<td>HDAM002L 1</td>
<td>HDAM002 L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SGIDE0A</td>
<td>HDAM001L 1</td>
<td>HDAM001 L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDX001G</td>
<td>HDAM001L 1</td>
<td>HDAM001 L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XSGIDE0A</td>
<td>HDAM002L 1</td>
<td>HDAM002 L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

**Figure 122. PSB Segment Reference report**

**Report field descriptions**
The meaning of each column is as follows:

**SEGMENT**
This field shows the segment name that specified by the SEGMENT option.

The following four columns are information about the DB PCB in which each SEGMENT is defined by the SENSEG NAME= statement of the PSBGEN utility.

**PSBNAME**
This field shows the PSB name which includes the DB PCB.

**PCB NO**
This field shows the sequential number assigned to the PCB in the PSB.

**REF DBD**
This field shows the DBD name that the DB PCB refers to by PCB DBDNAME= statement of the PSBGEN utility.

**PROCOPT**
This field shows the PCB PROCOPT that defined in the DB PCB.
**PSB PROCOPT reference reports for PSB and ACB(PSB)**

If a POPTREF control statement is specified, a PSB PROCOPT Reference report or an ACB(PSB) PROCOPT Reference report is generated. These reports contain information about the PSBs or PSB-type ACBs that match the criteria that is defined by the POPTREF control statement.

**Subsections:**
- “Sample report”
- “Report field descriptions”

**Sample report**

The following figure shows an example of the PSB PROCOPT Reference report.

![Sample report](image-url)

**Report field descriptions**

The meaning of each column is as follows:

**PSBNAMES**
- The name of the PSB that met the criteria that are defined by the POPTREF control statement.

**PCB NO**
- The sequential number that is assigned to the PCB in the PSB.

**REF DBD**
- The name of the DBD that is referenced by the PCB. This name is defined by the DBDNAME or the NAME parameter in the PCB statement of the PSBGEN utility.
The following columns show details about the PCB. Each row shows information from the PCB statement, the SENSEG statement, or both.

**SEGMENT**

The segment name in the DBD, which is referenced from the SENSEG statements of the PCB. This field is blank if the row is for a PCB statement.

**PCB PROCOPT**

The processing options (PROCOPT) that are defined in the PCB statement.

The PROCOPT values in the PCB statement are shown in the first row for each DBD regardless of whether the row is for a PCB statement or a SENSEG statement.

**SENSEG PROCOPT**

The processing options (PROCOPT) that are defined in the SENSEG statement.

This field is blank if the row is for a PCB statement.

**Unreferenced ACB(DBD) report**

If you specify the UNREF ACB control statement, the utility generates the Unreferenced ACB(DBD) report. This report contains information about DBD-type ACBs that are not referenced by any PSB-type ACBs in the ACBLIB.

**Sample report**

The following figure shows an example of the Unreferenced ACB(DBD) report. If all DBD-type ACBs are referenced by one or more PSB-type ACBs, NONE is printed.

---

**Library member list report for DBD or PSB**

The library member list report for DBD or PSB contains information about the members in the data sets that are concatenated to DBDLIB DD or PSBLIB DD. The information includes the IMS version that generated the DBD or PSB, generated date and time, and the size of the member record.

If a LISTLIB DBD control statement is specified, the DBD library member list report is generated. If a LISTLIB PSB control statement is specified, the PSB library member list report is generated.

**Subsections:**

- “Sample report: DBD library member list report” on page 324
- “Sample report: PSB library member list report” on page 324
- “Report field descriptions” on page 324
Sample report: DBD library member list report

The following figure shows an example of the DBD library member list report. This report provides information about the members in the data sets that are concatenated to DBDLIB DD. The DBDs are ordered alphabetically by DBD name.

<table>
<thead>
<tr>
<th>DBD Name</th>
<th>DS#</th>
<th>LVL</th>
<th>Size</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDIMAP00</td>
<td>001</td>
<td>12.1</td>
<td>2,112</td>
<td></td>
</tr>
<tr>
<td>CDPACT00</td>
<td>001</td>
<td>12.1</td>
<td>416</td>
<td></td>
</tr>
<tr>
<td>DSSCHIVN</td>
<td>002</td>
<td>13.1</td>
<td>2,096</td>
<td></td>
</tr>
<tr>
<td>HDAM1DSG</td>
<td>002</td>
<td>13.1</td>
<td>1,880</td>
<td></td>
</tr>
<tr>
<td>HDAM2DS</td>
<td>002</td>
<td>13.1</td>
<td>400</td>
<td></td>
</tr>
<tr>
<td>HDAM2DSA</td>
<td>002</td>
<td>13.1</td>
<td>400</td>
<td>ALIAS, NO DBDS REFERENCED</td>
</tr>
</tbody>
</table>

Figure 125. DBD library member list report

Sample report: PSB library member list report

The following figure shows an example of the PSB library member list report. This report provides information about the members in the data sets that are concatenated to PSBLIB DD. The PSBs are ordered alphabetically by PSB name.

<table>
<thead>
<tr>
<th>PSB Name</th>
<th>DS#</th>
<th>LVL</th>
<th>Size</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSBP000</td>
<td>002</td>
<td>13.1</td>
<td>328</td>
<td></td>
</tr>
<tr>
<td>PSBP010</td>
<td>002</td>
<td>13.1</td>
<td>1,328</td>
<td></td>
</tr>
<tr>
<td>PSBSMUUL</td>
<td>001</td>
<td>12.1</td>
<td>672</td>
<td></td>
</tr>
<tr>
<td>PSBSTUIL</td>
<td>001</td>
<td>12.1</td>
<td>272</td>
<td></td>
</tr>
<tr>
<td>PSBW01</td>
<td>002</td>
<td>12.1</td>
<td>152</td>
<td>NO DBDS REFERENCED</td>
</tr>
<tr>
<td>PSBW01A</td>
<td>002</td>
<td>12.1</td>
<td>152</td>
<td>ALIAS, NO DBDS REFERENCED</td>
</tr>
</tbody>
</table>

Figure 126. PSB library member list report

Report field descriptions

The meaning of each column is as follows:

**DS#** Each number indicates a DBD or PSB library that is specified in the DBDLIB DD or PSBLIB DD statement. The numbers are assigned by the utility.

**IMS LVL**

The IMS version that generated the DBD or PSB.

**SIZE (BYTES)**

The size of the member record.

**COMMENTS**

The following comments might be shown:
- ALIAS
- NO DBDS REFERENCED
If the SYSIN data set contains one or more DECODE or SEGREF control statements, the resulting MAPOUT data set contains the input to the DBD/PSB/ACB Mapper program.

The following figure shows an example of the input that is written to the MAPOUT data set.

```
DBD=HDAM01
DBD=HIDAM01
*  
PSB=PSB01
PSB=PSB02
```

*Figure 127. Input to the DBD/PSB/ACB Mapper program (MAPOUT data set)*

The OPTPRT data set contains the Run-time Option report, which shows the options that were applied at run time.

This report is printed when both of the following conditions are met:

- The OPTPRT DD statement is specified in JCL.
- One or more function control statements (except for SEGREF control statements) are specified in the SYSIN data set.

The following figure shows an example of the Run-time Option report.

```
<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>MEMBER</th>
<th>TYPE</th>
<th>KEYWORD</th>
<th>OPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECODE DBD@0001</td>
<td>DBD</td>
<td>DECOPT</td>
<td>COMPRESS=NO</td>
<td>COMMENT=NO</td>
</tr>
<tr>
<td>DECODE DBD@0002</td>
<td>DBD</td>
<td>DECOPT</td>
<td>COMPRESS=NO</td>
<td>COMMENT=NO</td>
</tr>
<tr>
<td>DECODE DBD@0003</td>
<td>DBD</td>
<td>DECOPT</td>
<td>COMPRESS=NO</td>
<td>COMMENT=NO</td>
</tr>
<tr>
<td>DECODE PSB@0001</td>
<td>PSB</td>
<td>DECOPT</td>
<td>COMPRESS=NO</td>
<td>COMMENT=NO</td>
</tr>
<tr>
<td>DECODE PSB@0002</td>
<td>PSB</td>
<td>DECOPT</td>
<td>COMPRESS=YES</td>
<td>COMMENT=NO</td>
</tr>
<tr>
<td>DECODE DBD@0004</td>
<td>DBD</td>
<td>DECOPT</td>
<td>COMPRESS=YES</td>
<td>COMMENT=NO</td>
</tr>
<tr>
<td>DECODE DBD@0005</td>
<td>DBD</td>
<td>DECOPT</td>
<td>COMPRESS=YES</td>
<td>COMMENT=NO</td>
</tr>
<tr>
<td>DECODE ACB@0001</td>
<td>ACB</td>
<td>DECOPT</td>
<td>COMPRESS=YES</td>
<td>COMMENT=YES</td>
</tr>
<tr>
<td>DECODE ACB@0002</td>
<td>ACB</td>
<td>DECOPT</td>
<td>COMPRESS=YES</td>
<td>COMMENT=YES</td>
</tr>
<tr>
<td>DECODE ACB@0003</td>
<td>ACB</td>
<td>DECOPT</td>
<td>COMPRESS=YES</td>
<td>COMMENT=YES</td>
</tr>
</tbody>
</table>
```

*Figure 128. Run-time Option report*
Use the Site Default Generation utility for the DBD/PSB/ACB Reversal utility (Reversal Site Default Generation utility) to set your own default values for the SYSIN control statements. This utility runs as a batch job.

The following topics describe how to use the Reversal Site Default Generation utility to generate and report SYSIN site default table.

**Reversal Site Default Generation utility overview**

The Reversal Site Default Generation utility has two functions; generating the SYSIN site default table, and reporting the contents of the SYSIN site default table.

**Subsections:**

- “Generating the SYSIN site default table”
- “Reporting the SYSIN site default table”

**Generating the SYSIN site default table**

The Reversal Site Default Generation utility analyzes the SYSIN control statements and generates a source code for the SYSIN site default table.

When the FABNRVRS program finds the name FABNCTL0 (the SYSIN site default table) in the STEPLIB libraries, the DBD/PSB/ACB Reversal utility loads the table and uses it as the default values of the SYSIN statement.

**Note:** The SYSIN site default table for the DBD/PSB/ACB Reversal utility is available only when the utility runs as a stand-alone utility. It is not available when the DBD/PSB/ACB Reversal utility is called from other components.

**Reporting the SYSIN site default table**

The Reversal Site Default Generation utility can read the SYSIN site default table and print the site default values in the reports.

**Setting site default values for the DBD/PSB/ACB Reversal utility**

To generate a site default table by using the Reversal Site Default Generation utility, you must code the Reversal Site Default Generation utility JCL, run the job, assemble and link-edit the source code, and concatenate the load module library to the DBD/PSB/ACB Reversal JCL.

**Procedure**

1. Run the Reversal Site Default Generation utility (FABNTGEN) job step to create source code of the SYSIN site default table (FABNCTL0).
   You can use sample JCL to run the utility. Locate member FABNDFL1 in the SHPSSAMP library and modify the sample JCL. The FABNDFL1 sample JCL creates a source code and then assembles and link-edits the source code. Therefore, if you use FABNDFL1, you can omit Step 2 on page 327.

See the following topics for additional information:

- For the format of the EXEC statement and the list of DD statements, see “JCL requirements for the Reversal Site Default Generation utility” on page 329.
• For a description of the control statements, see “Control statements for the Reversal Site Default Generation utility” on page 330.

2. Assemble and link-edit the FABNCTL0 source code.

   To create the SYSIN site default table module FABNCTL0, assemble and link the SYSPUNCH data set that is generated by FABNTGEN.

   For SYSIN of the assemble job step, specify the SYSPUNCH data set that is generated in the FABNTGEN processing. In the link-edit job step, consider using AMODE=31 and RMODE=ANY instead of the default values (which are AMODE=24 and RMODE=24) by adding AMODE=31 and RMODE=ANY to the EXEC statement PARM list.

3. Concatenate the load module library in which FABNCTL0 resides to the STEPLIB of the DBD/PSB/ACB Reversal FABNRVRS JCL.

   To use the site default table, the library for the SYSIN site default table module (FABNCTL0) must be concatenated to the STEPLIB DD of FABNRVRS runtime JCL.

   **Tip:** If you specify a value in the SYSIN control statement in the DBD/PSB/ACB Reversal FABNRVRS JCL, you can override the site default value at run time.

**Example**

The following figure shows a sample for creating the SYSIN site default table module FABNCTL0.
Figure 129. Sample JCL for creating the site default table module FABNCTL0
JCL requirements for the Reversal Site Default Generation utility

To run the Reversal Site Default Generation utility (FABNTGEN), supply an EXEC statement with the PARM parameters and appropriate DD statements.

Subsections:
- “EXEC statement”
- “DD statement summary”
- “DD statements”

EXEC statement

This statement must be in the following format:

```
// EXEC PGM=FABNTGEN,PARM='parameter'
```

Specify GEN or REPORT for parameter.

**GEN**  Specifies that the SYSIN site default table is generated. GEN is the default.

**REPORT**  Specifies that the site default values that are stored in the SYSIN site default table are printed.

Sample JCL streams that run the FABNTGEN program with PARM='GEN' and PARM='REPORT' are in the SHPSSAMP data set. The member names are FABNDFL1 and FABNDFL2.

DD statement summary

The following table summarizes the DD statements.

<table>
<thead>
<tr>
<th>DD name</th>
<th>Use</th>
<th>Format</th>
<th>EXEC PARM=</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>PARM='GEN'</td>
</tr>
<tr>
<td>STEPLIB</td>
<td>Input</td>
<td>PDS</td>
<td>Required</td>
</tr>
<tr>
<td>SYSIN</td>
<td>Input</td>
<td>LRECL=80</td>
<td>Required</td>
</tr>
<tr>
<td>SYSPUNCH</td>
<td>Output</td>
<td>LRECL=80</td>
<td>Required</td>
</tr>
<tr>
<td>SYSOUT</td>
<td>Output</td>
<td>LRECL=133</td>
<td>Required</td>
</tr>
<tr>
<td>SYSABEND or</td>
<td>Output</td>
<td>LRECL=133</td>
<td>Optional</td>
</tr>
<tr>
<td>SYSUDUMP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DD statements

Code the following DD statements to identify the source of input and the placement of output information:

**STEPLIB DD**

This required input data set contains the IMS Library Integrity Utilities load module library. When PARM='REPORT' is specified in the EXEC statement, you must also specify the data set that includes the SYSIN site default table module member (FABNCTL0).
SYSIN DD
This data set is required when PARM='GEN' is specified in the EXEC statement. The format is the same as the FABNRVRS SYSIN statement. Specify this input data set to include your own default values for the SYSIN control statements.

SYSPUNCH DD
This output data set is required when PARM='GEN' is specified in the EXEC statement. An assembler source code of the SYSIN site default table is produced in this data set. The following DCB parameters must be specified:

- RECFM=F or FB
- LRECL=80
- BLKSIZE=80 or multiple of 80

SYSOUT DD
This output data set is required. The messages and the echo of the SYSIN control statements in the SYSIN data set, which are issued by the FABNTGEN, are printed in this data set. You can specify SYSOUT=* (or JES output class name) instead of a data set name.

SYSUDUMP DD (or SYSABEND)
This data set defines the output for the system ABEND dump routine. This DD statement is used only when a dump is required.

Control statements for the Reversal Site Default Generation utility
The SYSIN control statements are required to generate the SYSIN site default table. You can change some of the default values of the DBD/PSB/ACB Reversal utility to your site-specific values by specifying the appropriate options and operands.

The Reversal Site Default Generation utility analyzes the runtime option control statements and sets the site default values. Other control statement keywords, such as DECODE, are ignored. If the keywords are omitted, the DBD/PSB/ACB Reversal utility system default values will be used.

OPTION
This control statement keyword is required if no DECOPT statement is specified.

DECOPT
This control statement keyword is required if no OPTION statement is specified.

The following table shows the runtime options that are available for these control statement keywords.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Option (abbreviations)</th>
<th>Operand</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTION</td>
<td>ACB_GSAM</td>
<td>YES or NO</td>
</tr>
<tr>
<td>OPTION</td>
<td>ACB_REFERENCED</td>
<td>YES or NO</td>
</tr>
<tr>
<td>OPTION</td>
<td>PGM_COBOL</td>
<td>YES or NO</td>
</tr>
<tr>
<td>DECOPT</td>
<td>CHECK_LEN</td>
<td>YES or NO</td>
</tr>
<tr>
<td>DECOPT</td>
<td>COMMENT(C)</td>
<td>YES or NO</td>
</tr>
<tr>
<td>DECOPT</td>
<td>COMPRESS(COMP)</td>
<td>YES or NO</td>
</tr>
</tbody>
</table>
Table 23. Options for the site default in the SYSIN control statement (continued)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Option (abbreviations)</th>
<th>Operand</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECOPT</td>
<td>PCB_LABEL</td>
<td>YES or NO</td>
</tr>
<tr>
<td>DECOPT</td>
<td>SENSEG_PROCOPT</td>
<td>YES or NO</td>
</tr>
<tr>
<td>DECOPT</td>
<td>VERSION_GENDATE</td>
<td>YES or NO</td>
</tr>
</tbody>
</table>

For descriptions of the keyword and the options, see “Control statements for the DBD/PSB/ACB Reversal utility” on page 288.

Output from the DBD/PSB/ACB Reversal Site Default Generation utility

The SYSOUT data set contains the output from the DBD/PSB/ACB Reversal Site Default Generation utility.

SYSOUT data set

The SYSOUT data set contains activity logs and error messages. When PARM='REPORT' is specified on the EXEC statement parameter, in addition to the activity logs and error messages, the site default values that are stored in the SYSIN site default table are printed in this data set.

The following figures show messages that are generated in the SYSOUT data set.

The following messages are printed when the EXEC PARM parameter is 'GEN'.

---

Figure 130. Messages in the SYSOUT data set when PARM='GEN'

The following messages are printed when the EXEC PARM parameter is 'REPORT'.

---

Figure 131. Messages in the SYSOUT data set when PARM='REPORT'
Chapter 9. Catalog Manager utility

The Catalog Manager utility helps you ensure that the DBDs and PSBs in the IMS catalog match the DBDs and PSBs in the ACB libraries.

Topics:
- “Catalog Manager utility overview” on page 334
- “Catalog Manager utility restrictions” on page 336
- “Validating DBDs and PSBs in the IMS catalog” on page 337
- “JCL requirements for the Catalog Manager utility” on page 338
- “Control statements for the Catalog Manager utility” on page 342
- “Output from the Catalog Manager utility” on page 344
Catalog Manager utility overview

The Catalog Manager utility helps you ensure that the DBDs and PSBs in the IMS catalog are maintained correctly. It validates that the DBDs and PSBs in the ACB libraries match the DBDs and PSBs in the IMS catalog. The utility generates several reports, including the IMS Catalog Validation report, which contains the results of the validation process.

Subsections:
- “Function overview”
- “Program structure and job step”
- “Data flow”

Function overview

The Catalog Manager utility checks whether the time stamps of DBDs and PSBs in the IMS catalog match the ACBGEN time stamps (time stamps added by the Application Control Block Generation utility) of DBD-type ACB and PSB-type ACB members in the ACB libraries. The utility also checks the consistency of the following information:

- When database versioning is enabled, the utility checks whether the database version number is the same for each DBD.
- When IMS loads ACBs from the IMS catalog instead of from ACB libraries, the utility checks whether the time stamps of DBD-type ACBs and PSB-type ACBs in the IMS catalog directory match the time stamps of DBDs and PSBs in the IMS catalog database.

Whenever the DBDs and PSBs in the ACB libraries are updated, the IMS catalog must be populated during or after the ACBGEN process. When the IMS catalog is populated, information about the most recent DBDs and PSBs in the ACB libraries is replicated to the IMS catalog. If DBDs and PSBs are maintained correctly in the IMS catalog, the time stamps of DBDs and PSBs in the IMS catalog always match the ACBGEN time stamps of the DBD and PSB members in the ACB libraries.

The IMS catalog can contain more than one instance for each DBD and PSB. If multiple instances exist, the utility refers to the most recent instance and uses the time stamp and the database version number from that instance for validation.

By validating the time stamps and database version numbers with the Catalog Manager utility, you can ensure that the DBDs and PSBs that are stored in the IMS catalog are maintained correctly.

Program structure and job step

The Catalog Manager utility is provided as an MVS batch utility program. The utility consists of one program, FABXCATM, which controls other load modules. This program uses an input format that is specified in the FABXCLN data set.

While the utility is running, WTO messages on the console show program processing status. Reports and error messages are written in the data sets that are defined by the FABXCRP0 and FABXCRP1 DD statements.

Data flow

The following figure shows the general data flow for the Catalog Manager utility.
The input consists of the FABXCIN control data set (contains the control statements), ACBLIB data sets, and the IMS catalog. The output consists of reports and messages.

Figure 132. Data flow for the Catalog Manager utility
Catalog Manager utility restrictions

Certain restrictions apply when you use the Catalog Manager utility.

The Catalog Manager utility has the following restrictions:

- The Catalog Manager utility supports only IMS Version 12 and later because the IMS catalog function is supported only in those versions.

- The Catalog Manager utility checks only DBDs and PSBs that are found in the ACB libraries. DBDs and PSBs that are found in the IMS catalog but not in the ACB libraries are not checked or included in the IMS Catalog Validation report.

- Depending on the maintenance level of IMS, the Catalog Manager utility cannot validate PSBs that contain PCBs referring to a GSAM or a logical database. This is because no time stamp information is stored for such PSBs in the IMS catalog. To validate such PSBs generated by IMS Version 13, you must apply APAR PI27237 to IMS.

- When IMS loads ACBs from the IMS catalog instead of from the ACB libraries, the Catalog Manager utility checks the time stamps in the current ACBs. The utility does not check the time stamps in pending ACBs.
Validating DBDs and PSBs in the IMS catalog

To validate DBDs and PSBs in the IMS catalog by using the Catalog Manager utility, you must prepare the JCL for the Catalog Manager utility and submit the job.

About this task

Sample JCL for the Catalog Manager utility is in the SHPSJCL0 library, member FABXCIYP. You can modify this sample JCL and then use it to run the utility.

Procedure

1. In the Catalog Manager utility JCL, code the EXEC statement and DD statements.
   See “JCL requirements for the Catalog Manager utility” on page 338.
2. In the FABXCIN data set, code the control statements for the Catalog Manager utility.
   See “Control statements for the Catalog Manager utility” on page 342.
3. Submit the job.
4. Check the job-step return code, WTO messages, and output data sets that are generated. The validation result is in the IMS Catalog Validation report in the FABXCRP1 data set.
   See “Output from the Catalog Manager utility” on page 344.

What to do next

If inconsistencies were found by the validation process, run the ACB Generation and Catalog Populate utility (DFS3UACB) or the IMS Catalog Population utility (DFS3PU00) to populate the IMS catalog.
JCL requirements for the Catalog Manager utility

When you code the JCL to run the Catalog Manager utility, include the EXEC statement and appropriate DD statements.

Subsections:
- “JCL example”
- “EXEC statement” on page 339
- “DD statements” on page 339

JCL example

The following figure shows an example of the JCL that is required to run the Catalog Manager utility program.

```
//CATMANJ JOB ....
//STEP EXEC PGM=FABXCATM
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMS13.SDFSRESL,DISP=SHR
//DFSRESLB DD DSN=IMS13.SDFSRESL,DISP=SHR
//IMS DD DSN=PROD.PSBLIB,DISP=SHR
// DD DSN=PROD.DBDLIB,DISP=SHR
//DFSVSAMP DD *
0512,9
1024,9
2048,9
4096,9
16384,9
32768,9
10BF=(2048,4,N,N)
10BF=(4096,4,N,N)
10BF=(8192,4,N,N)
10BF=(32000,4,N,N)
/*
//PROCLIB DD DSN=IMS13.PROCLIB
//RECON1 DD DSN=PROD.RECON1,DISP=SHR
//RECON2 DD DSN=PROD.RECON2,DISP=SHR
/*/n
//ACBLIB DD DSN=PROD.ACBLIB
//FABXPPRM DD *
DLI,FABXCATM,DFSPL00,,,,,,,,,,Y,N,,,,,,,,,,DFSDF=CAT
//FABXCIN DD *
   PROC FUNC=CHECK_GENTIME
   DBD NAME**
   PSB NAME**
   END
/*
```

Figure 133. Example of JCL for the Catalog Manager utility

In this example, it is assumed that the DFSDFxxx PROCLIB member and RECON data sets were used to configure the environment for the IMS catalog. When you code the JCL, add appropriate DD statements based on how you configured the IMS catalog environment. This applies, for example, if you used the Catalog Definition exit routine (DFS3CDX0) or the IMS catalog partition definition data set (DFSHDBSC) to configure the environment for the IMS catalog.
EXEC statement

The EXEC JCL statement must specify the FABXCATM program. No PARM operand is required.

//stepname EXEC PGM=FABXCATM

You can specify IMSplex name and a group of DBRC instances to access the RECON data set. Here is an example of the statement:

//stepname EXEC PGM=FABXCATM,PARM='IMSPLEX=imsplex, DBRCGRP=dbrcgrp'

IMSPLEX=imsplex
A 1 - 5 character IMSplex name used for RECON data sets.

DBRCGRP=dbrcgrp
A 1 - 3 character identifier (ID) assigned to a group of DBRC instances that access the same RECON data set in an IMSplex.

DD statements

Code the following DD statements to identify the source of input and the placement of output information:

STEPLIB DD or JOBLIB DD
Required input data set. Specify the LIU load module library data set, which contains the Catalog Manager utility, and the IMS.SDFSRESL data set.

Optionally, specify the following resources:
• If you use the Catalog Definition exit routine (DFS3CDX0), specify the load module data set that contains the exit routine.
• If you want the RECON data sets or the DFSHDBSC data set to be dynamically allocated, specify the MDA library.
• If you use the SCI exit routine for your IMS environment, specify the load module data set that contains the exit routine.

DFSRESLB DD
Required input data set. Specify the IMS.SDFSRESL data set.

IMS DD
Required input data set. Specify the PSB and DBD libraries that contain the DBDs and PSBs for the IMS catalog.

DFSVSAM DD
Required input data set. Specify the buffer pool parameters data set.

RECON1 DD
RECON2 DD
RECON3 DD
Optional input data sets. Specify the RECON data sets if the IMS catalog database is registered in the RECON data sets.

PROCLIB DD
Optional input data set. Specify the IMS.PROCLIB data set that contains the DFSDFxxx member if a DFSDFxxx member is used for the IMS catalog.

DFSHDBSC DD
Optional input data set. Specify the IMS catalog partition definition data set (DFSHDBSC) if the IMS catalog was defined with the IMS Catalog Partition Definition Data Set utility (DFS3UCD0).
**ACBLIB DD**
Required input data set. Specify one or more ACB libraries that contain the DBD and PSB members to check.

**FABXPPRM DD**
Optional input statement. Specify the parameters for the DL/I batch program that the Catalog Manager utility uses to read the IMS catalog database.

The parameters must include the name of the DFSDFxxx PROCLIB member that contains the processing options for the IMS catalog. Here is an example of the parameters:

```
//FABXPPRM DD *
DLI,FABXCATM,DFSCPL00,,,, Y,,Y,......DFSDF=CAT
/*
```

You can specify any value for the first, second, and third positional parameters because the Catalog Manager utility supplies appropriate values for these parameters.

The format of the parameter is the same as the DFS3PPRM DD statement for the ACB Generation and Catalog Populate utility (DFS3UACB). You can reuse the parameters that you specify in the DFS3PPRM data set.

**Related reading:** For more information, see the topic "ACB Generation and Catalog Populate utility (DFS3UACB)" in *IMS System Utilities*.

You can specify the IMSPlex name and the group of DBRC instances that access the RECON data set to the FABXPPRM DD statement. Here is an example of the parameters:

```
//FABXPPRM DD *
DLI,FABXCATM,DFSCPL00,,,, ,Y,,N,......imsplex,,DFSDF=CAT,DBRCGRP=dbrcgroup
/*
```

If you specify the IMSPlex name and the group of DBRC instances on both the EXEC statement and the FABXPPRM DD statement, the parameters on the FABXPPRM DD statement are used.

If the Catalog Definition exit routine (DFS3CDX0) is used, you can omit the FABXPPRM DD statement. If DFS3CDX0 is not used and you omit the FABXPPRM DD statement, the Catalog Manager utility uses the following statement as the default:

```
//FABXPPRM DD *
DLI,FABXCAPL,DFSCP000,,,, ,Y,,N,......DFSDF=CAT
/*
```

**FABXCN DD**
Required input data set. Specify the data set that contains the control statements for the Catalog Manager utility. The DCB parameters must be RECFM=FB, LRECL=80, and BLKSIZE must be a multiple of 80.

**Related reading:** For information about the control statements of the Catalog Manager utility, see "Control statements for the Catalog Manager utility" on page 342.

**FABXCRPO DD**
**FABXCRP1 DD**
Optional output data sets. Specify the data sets for the Catalog Manager utility reports. The DCB parameters must be RECFM=FBA, LRECL=133, and BLKSIZE must be a multiple of 133.
If the DD statements are not specified, the Catalog Manager utility allocates SYSOUT=* to the DD statements and generates the reports.

**Related reading:** For information about the reports that are generated by the Catalog Manager utility, see “Output from the Catalog Manager utility” on page 344.

**SYSUDUMP DD, SYSABEND DD, or SYSMDUMP DD**
Optional output data sets. Define dump data sets.
Control statements for the Catalog Manager utility

The control statements for the Catalog Manager utility are defined in the FABXCIN data set.

Subsections:
- “Control statement example”
- “Syntax rules”
- “Statements, keywords, and parameters”

Control statement example

The following figure shows an example of the FABXCIN control statements.

```plaintext
//FABXCIN
DD *
PROC FUNC=CHECK_GENTIME *
   DBD NAME=HDAMDB1
   PSB NAME=* *
END
/*
```

Figure 134. Control statement example

Syntax rules

The control statements for the Catalog Manager utility must adhere to the following syntax rules:
- Control statements can be coded on any columns in the range of 2 - 80.
- A statement with an asterisk (*) in column 1 is treated as a comment.
- Each statement consists of a statement type, a keyword, and a parameter as follows:

  statement-type keyword=parameter

Statements, keywords, and parameters

PROC statement

Required statement. This statement must be coded on the first line with the following keyword and parameter:

  FUNC=CHECK_GENTIME

Specifies that you want the utility to compare the ACBGEN time stamp of each DBD and PSB member in the ACB libraries to the time stamp of the corresponding DBD and PSB resource in the IMS catalog.

DBD statement

Optional statement. Use this statement to validate specific DBDs. Specify this statement with the following keyword and parameter:

  NAME=resource_name

Specify a DBD name. You can use wildcard characters to create a pattern-matching expression that specifies more than one DBD.
**PSB statement**

Optional statement. Use this statement to validate specific PSBs. Specify this statement with the following keyword and parameter:

**NAME=resource_name**

Specify a PSB name. You can use wildcard characters to create a pattern-matching expression that specifies more than one PSB.

**Tips for using wildcard characters**

To specify multiple DBDs or PSBs, specify a wildcard in any position of a character string. The asterisk (*) and the percent sign (%) are supported as wildcard characters. An asterisk represents 0 - 8 characters, and a percent sign represents a single character. If two or more asterisks are specified sequentially, only the first asterisk is recognized.

For example, you can specify the wildcard characters in the following ways:

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Coding example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validate all DBD members in the ACB libraries</td>
<td>DBD NAME**</td>
</tr>
<tr>
<td>Validate DBDs that have a name that begins with the letter H</td>
<td>DBD NAME=H*</td>
</tr>
<tr>
<td>Validate PSBs that have a name that begins with the letters ABC, have any letter as the fourth character, and contain 001 as the fifth to seventh characters</td>
<td>PSB NAME=ABC%001</td>
</tr>
</tbody>
</table>

**END statement**

Optional statement. Use this statement to indicate the end of the control statements.
Output from the Catalog Manager utility

Output from the Catalog Manager utility includes the FABXCRP0 data set and the FABXCRP1 data set.

FABXCRP0 data set

The FABXCRP0 data set contains the Control Statement report, which shows the echo of the FABXCIN control statements and the selected runtime options.

<table>
<thead>
<tr>
<th>PROC</th>
<th>FUNCTION</th>
<th>CHECK_GENTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROC</td>
<td>FUNC</td>
<td>CHECK_GENTIME</td>
</tr>
</tbody>
</table>

* Check all DBD-type & PSB-type resources.

<table>
<thead>
<tr>
<th>DBD</th>
<th>NAME</th>
<th>**</th>
<th>PSB</th>
<th>NAME</th>
<th>**</th>
</tr>
</thead>
<tbody>
<tr>
<td>END</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

FABXCRP1 data set

The FABXCRP1 data set contains the FABXPPRM Echo Back report, the IMS Catalog Environment report, and the IMS Catalog Validation report.

Subsections:

- "FABXPPRM Echo Back report"
- "IMS Catalog Environment report"
- "IMS Catalog Validation report" on page 345

FABXPPRM Echo Back report

This report contains an echo of the FABXPPRM parameters.

| DL1, FABXCAPL, OFSCP000, | M, N, | DFSDF=HGN |

IMS Catalog Environment report

This report contains environment information about the IMS system and the IMS catalog.
This report contains the following fields:

**IMS ID**
IMS ID.

**IMS VERSION**
IMS version.

**DFSDF MEMBER NAME**
DFSDF PROCLIB member name. This field is blank if the DFS3CDX0 exit routine was used.

**DFS3CDX0 ROUTINE**
Whether the DFS3CDX0 exit routine was used. YES indicates that the DFS3CDX0 was used. Blank indicates that the DFSDF member was used.

**ACB MANAGEMENT**
The location from which the ACBs were loaded. The following locations are used:

**CATALOG**
ACBs were loaded from the IMS catalog.

**ACBLIB**
ACBs were loaded from the ACB libraries.

**ALIAS**
The alias name of the IMS catalog.

**CATALOG HLQ**
The high-level qualifier (HLQ) of the IMS catalog.

**IMS Catalog Validation report**

This report contains validation results for the DBDs and PSBs. If ACBs were loaded from the ACB libraries, the report contains validation results for the DBDs and PSBs in the ACB libraries and the IMS catalog database. If ACBs were loaded from the IMS catalog, the report contains validation results for the DBDs and PSBs in the ACB libraries, the IMS catalog database, and the IMS catalog directory.

The following figure shows an example of the report when ACBs were loaded from ACB libraries.
This report contains the following fields:

**NAME**

The resource name. Either the name of the DBD or PSB.

**VALIDATION RESULT**

The result of the validation. The following indicators are used:

- **VALID**
  - The time stamp and the database version number in the IMS catalog match the ACBGEN time stamp and the database version number of the member in the ACB library.

- **INVALID**
  - One or more of the following conditions apply:
    - The time stamp in the IMS catalog does not match the ACBGEN time stamp of the member in the ACB library.
    - The database version number in the IMS catalog does not match the database version number in the ACB library.
    - The DBD or PSB is not contained in the IMS catalog.

To populate the IMS catalog and make the time stamps match, run the ACB Generation and Catalog Populate utility (DFS3UACB) or the IMS Catalog Population utility (DFS3PU00).

- **FAILED**
  - One or more of the following conditions met:
    - The time stamp of the member in the IMS catalog database does not match the time stamp of that member in the IMS catalog directory.
    - The Catalog Manager utility cannot validate the resource because the time stamp information was not found in the IMS catalog.

---

## DBD

<table>
<thead>
<tr>
<th>NAME</th>
<th>VALIDATION</th>
<th>ACBLIB</th>
<th>ACBGEN TIMESTAMP</th>
<th>CATALOG</th>
<th>TIMESTAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOD70A</td>
<td>INVALID</td>
<td>0000000001</td>
<td>2015/06/16 13:45:11.20</td>
<td>2015/06/16 13:45:11.20</td>
<td></td>
</tr>
<tr>
<td>DOD70B</td>
<td>VALID</td>
<td>2147483647</td>
<td>2015/06/16 13:45:11.20</td>
<td>2147483647</td>
<td>2015/06/16 13:45:11.20</td>
</tr>
<tr>
<td>DOD71A</td>
<td>VALID</td>
<td>0000000000</td>
<td>2015/06/16 13:45:11.20</td>
<td>0000000000</td>
<td>2015/06/16 13:45:11.20</td>
</tr>
<tr>
<td>DOD72A</td>
<td>VALID</td>
<td>0000000000</td>
<td>2015/06/16 13:45:11.20</td>
<td>0000000000</td>
<td>2015/06/16 13:45:11.20</td>
</tr>
<tr>
<td>DOD73A</td>
<td>VALID</td>
<td>0000000000</td>
<td>2015/06/16 13:45:11.20</td>
<td>0000000000</td>
<td>2015/06/16 13:45:11.20</td>
</tr>
<tr>
<td>DOD74A</td>
<td>VALID</td>
<td>0000000000</td>
<td>2015/06/16 13:45:11.20</td>
<td>0000000000</td>
<td>2015/06/16 13:45:11.20</td>
</tr>
</tbody>
</table>

## PSB

<table>
<thead>
<tr>
<th>NAME</th>
<th>VALIDATION</th>
<th>ACBLIB</th>
<th>ACBGEN TIMESTAMP</th>
<th>CATALOG</th>
<th>TIMESTAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB009</td>
<td>INVALID</td>
<td>2015/06/16 13:45:13.75</td>
<td>2015/06/16 13:45:10.26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSB001</td>
<td>INVALID</td>
<td>2015/06/16 13:45:07.09</td>
<td>2015/06/16 13:45:11.20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSB003</td>
<td>VALID</td>
<td>2015/06/16 13:45:12.34</td>
<td>2015/06/16 13:45:12.34</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Figure 138. Example of the IMS Catalog Validation report
Run the ACB Generation and Catalog Populate utility (DFS3UACB) or the IMS Catalog Population utility (DFS3PU00) to fix the condition.

The FAILED indicator is also shown for a PSB that contains PCBs that refer to a GSAM database or a logical database. See "Catalog Manager utility restrictions" on page 336.

**ACBLIB**
For a DBD, the database version number and the ACBGEN time stamp of the DBD member in the ACB library. When either of the following conditions is met, a string of zeros is shown as the database version number:

- Database versioning is enabled and this DBD is the first version of the database.
- Database versioning is not enabled.

For a PSB, the ACBGEN time stamp of the PSB member in the ACB library.

**CATALOG**
For a DBD, the database version number and the time stamp obtained from the DBD in the IMS catalog. When either of the following conditions is met, a string of zeros is shown as the database version number:

- Database versioning is enabled and this DBD is the first version of the database.
- Database versioning is not enabled.

For a PSB, the time stamp obtained from the PSB in the IMS catalog.

When a DBD or PSB has multiple instances in the IMS catalog, the information is obtained from the most recent instance.

**DIRECTORY**
The time stamp obtained from the DBD-type ACB or PSB-type ACB in the IMS catalog directory.
Chapter 10. Advanced Application Control Block Generator utility

The Advanced Application Control Block Generator utility (also referred to as Advanced ACBGEN utility) is a functional replacement for the IMS Application Control Blocks Maintenance utility with enhancements.

Topics:

- “Advanced ACBGEN utility overview” on page 350
- “Generating application control blocks” on page 351
- “Merging Advanced ACBGEN load modules into the IMS SDFSRESL library” on page 352
- “Using the Advanced ACBGEN utility in an ACB Generation and Catalog Populate utility job” on page 354
- “JCL requirements for the Advanced ACBGEN utility” on page 355
- “Control statements for the Advanced ACBGEN utility” on page 359
- “Output from the Advanced ACBGEN utility” on page 363
Advanced ACBGEN utility overview

Advanced ACBGEN utility is a replacement for the IMS ACBGEN utility (DFSUACB0), with enhancements. The Advanced ACBGEN utility can also replace the IMS ACBGEN utility used to generate ACB members within ACB Generation and Catalog Populate utility (DFS3UACB) jobs. It uses some of the modules provided by IMS, and replaces others. Additionally, several utilities are provided to display and audit the contents of an ACB library.

The current ACB generation process provided with IMS consists of three basic modules. Each set of basic modules consists of a primary module and a number of supporting modules. In this information, only the primary module is discussed.

- The first basic module is the IMS ACBGEN utility, DFSUACB0. Its primary function is to prepare the ACB library for processing, and build a list of PSB names, and pass them, one at a time, to the Block Builder module.
- The second basic module is the Block Builder, DFSDLBL0. Its primary function is to load a PSB and its referenced DBDs and build DLI control blocks. These control blocks are then passed to the Block Mover module.
- The third basic module is Block Mover, DFSUAMB0. Its primary function is to take the DLI control blocks passed to it and build a PSB and one or more DMBs. The PSB is then written into the ACB library. The DMBs might or might not be written into the ACB library.

The Advanced ACBGEN utility replaces the IMS provided DFSUACB0 module and the related supporting modules. However, the DFSDLBL0 Block Builder and DFSUAMB0 Block Mover modules are used to build the PSBs and DMBs.

The Advanced ACBGEN utility builds a list of PSBs. The names of these PSBs are passed to the Block Builder, DFSDLBL0, where they are processed. Statistics and IMS generated messages for each PSB and DMB built are captured. When the PSB list has been processed, several reports are generated that provide a summary list of the ACB library members that have been added, deleted, replaced, or not replaced.

An important feature of Advanced ACBGEN utility is its management and presentation of the DFSnnnn messages generated during the ACBGEN process. The utility presents PSB and DBD information in a concise tabular format rather than the prose format used by the utility provided in IMS. As more PSBs and DBDs are involved in the ACBGEN, the significance of this presentation method increases.

The elapsed time can be reduced by reducing both the number of DASD EXCPs and the CPU time. As more PSBs are generated, the time is reduced, especially when a 'BUILD PSB=ALL' ACB is generated.

Advanced ACBGEN utility JCL is compatible with the IMS utility. The use of Advanced ACBGEN utility, whether it is used alone or called within an ACB Generation and Catalog Populate utility job, can be enabled by adding the load module library that includes the Advanced ACBGEN load modules to the top of the STEPLIB DD concatenation. Additional optional DD statements can be added to request Advanced ACBGEN utility features. These additional DD statements are ignored if they are present when the IMS utility is used.
Generating application control blocks

To generate application control blocks by using the Advanced ACBGEN utility, you must prepare the JCL for the Advanced ACBGEN utility and submit the job.

About this task

Sample JCL for the Advanced ACBGEN utility is in the SHPSJCL0 library, member FABQIVP. You can modify this sample JCL and then use it to run the utility.

Procedure

1. In the Advanced ACBGEN utility JCL, code the EXEC statement and DD statements.
   You can modify the JCL that is used for the IMS ACBGEN utility.
   For the format of the EXEC statement and the list of DD statements, see “JCL requirements for the Advanced ACBGEN utility” on page 355.
2. Code the control statements for Advanced ACBGEN utility in the SYSIN data set and optionally in the ACBSYSIN data set.
   See “Control statements for the Advanced ACBGEN utility” on page 359.
3. Submit the job.
4. Check the output data sets that are generated.
   See “Output from the Advanced ACBGEN utility” on page 363.
Merging Advanced ACBGEN load modules into the IMS SDFSRESL library

If you do not want to modify IMS ACBGEN utility JCL, an alternative is to merge the Advanced ACBGEN load modules (FABQ*) into the IMS SDFSRESL library. However, this method is not recommended because this method requires extra steps when you install PTFs.

About this task

**Important:** When you merge the modules by completing this task, extra steps are required when you install PTFs to IMS Library Integrity Utilities or to IMS. To prevent any accidents from happening during future PTF installation, consider modifying STEPLIB (as in step 1 in “Generating application control blocks” on page 351) instead of merging the modules.

To use Advanced ACBGEN utility instead of the IMS ACBGEN utility, the LIU load module must precede the IMS SDFSRESL library in the STEPLIB in IMS ACBGEN utility JCL. When specified so, because alias name DFSUACB0 (IMS ACBGEN utility module name) is assigned to program load module FABQMAIN (LIU load module), Advanced ACBGEN utility starts instead of the IMS ACBGEN utility.

**Attention:**

- Do not accept USERMOD provided by FABQUMD3. After USERMOD is applied successfully, the IMS ACBGEN utility cannot be invoked unless the USERMOD is restored.
- If you need to apply a SYSMOD to the IMS DFSRRA80 module, restore USERMOD in advance and apply it again after applying the IMS SYSMOD.
- Whenever you apply USERMOD, rewrite the prerequisite SYSMOD statements, which can be referred to by the FABQUMD2 job, on the PRE operand in the FABQUMD3 JCL statement.
- If you need to apply a SYSMOD to IMS Library Integrity Utilities, run the FABQUMD1 job and merge the members again.

**Procedure**

Run the following steps to merge Advanced ACBGEN load modules into the IMS SDFSRESL library. The jobs are provided in the SHPSJCL0 JCL library.

1. Run the FABQUMD1 job.
   This job deletes alias DFSUACB0 from the IMS LIU SHPSLMD0 library and the LMOD entry of IMS LIU SMP/E CSI.

2. Run the FABQUMD2 job.
   This job runs SMP/E LIST of the IMS DFSRRA80 source entry. This information is required for the following FABQUMD3 job.

3. Run the FABQUMD3 job.
   This job runs SMP/E RECEIVE/APPLY of USERMOD to modify the IMS DFSRRA80 module so that the module invokes the IMS LIU FABQMAIN module instead of DFSUACB0.

4. Merge Advanced ACBGEN load modules in the target library SHPSLMD0 into the IMS SDFSRESL library.
What to do next

After running all of the jobs and merging the modules, you can run the Advanced ACBGEN utility without modifying the JCL statements for the IMS ACBGEN utility. Only for this case, the DFSRESLB DD statement can be omitted.
Using the Advanced ACBGEN utility in an ACB Generation and Catalog Populate utility job

To generate ACB members by using the Advanced ACBGEN utility in an ACB Generation and Catalog Populate utility job, modify the ACB Generation and Catalog Populate utility JCL so that the Advanced ACBGEN utility is used, and submit the job.

Procedure

1. In the ACB Generation and Catalog Populate utility (DFS3UACB) JCL, modify the DD statements so that the Advanced ACBGEN utility is used to generate ACB members.
   
   See “JCL requirements for the Advanced ACBGEN utility” on page 355.

2. Code the control statement for the Advanced ACBGEN utility in the SYSIN data set and, optionally, in the ACBSYSIN data set.

   See “Control statements for the Advanced ACBGEN utility” on page 359.

3. Submit the job.

4. Check the output data sets that are generated.

   See “Output from the Advanced ACBGEN utility” on page 363.
JCL requirements for the Advanced ACBGEN utility

JCL for running the Advanced ACBGEN utility, whether the utility is used alone or called within an ACB Generation and Catalog Populate utility (DFS3UACB) job, must meet JCL requirements.

The Advanced ACBGEN utility is JCL-compatible with the IMS ACBGEN utility (DFSUACB0). The Advanced ACBGEN utility supports a few unique DD statements, but most of the DD statements are common between the utilities.

This topic describes only the DD statements that require special attention. For a complete information about the JCL requirements of IMS utilities, see the following topics in *IMS System Utilities*:

- For the JCL requirements for the IMS ACBGEN utility, see the topic "Application Control Blocks Maintenance utility".
- To generate ACB members by using the Advanced ACBGEN utility in an ACB Generation and Catalog Populate utility job, see the JCL requirements for that utility in the topic "ACB Generation and Catalog Populate utility (DFS3UACB)".

If you run the IMS ACBGEN utility with a JCL stream that contains the Advanced ACBGEN utility unique DD statements, such as ACBSYSIN and DFSPRINT, those DD statements are ignored.

Subsections:

- "EXEC statement"
- “Common DD statements that are used differently” on page 356
- “Common DD statements” on page 356
- “DD statements used in ACB Generation and Catalog Populate utility (DFS3UACB) jobs” on page 357
- “Unique DD statements for the Advanced ACBGEN utility” on page 357

EXEC statement

The EXEC JCL statement does not need to be changed. You can specify the same EXEC JCL statement as the IMS ACBGEN utility. The first part of the EXEC statement must be in the form:

```
PGM=DFSRRC00
```

The parameter field must be in the form:

```
PARM=’UPB,PRECOMP,POSTCOMP’
```

**UPB** Indicates that the block maintenance utility is to receive control. This parameter is required.

**PRECOMP** Requests the IMS.ACBLIB data set be compressed before blocks are built.

**POSTCOMP** Requests compression after the blocks are built.

PRECOMP and POSTCOMP are optional and can be used in any combination.

The format of the EXEC JCL statement is the same when you use the Advanced ACBGEN utility in ACB Generation and Catalog Populate utility (DFS3UACB) jobs.
Common DD statements that are used differently

The following required DD statements are common to the Advanced ACBGEN utility, IMS ACBGEN utility, and the ACB Generation and Catalog Populate utility (DFS3UACB). However, these DD statements are used differently when they are specified for the Advanced ACBGEN utility.

STEPLIB DD
A STEPLIB DD statement or a JOBLIB DD statement must be provided. The data set name specified on this DD statement must be the name of the load module library that contains the Advanced ACBGEN utility.

The IMS RESLIB must be concatenated behind the DD statement that contains the Advanced ACBGEN utility.

DFSRESLB DD
This statement must be provided regardless of the APF-authorization of the STEPLIB DD. This statement points to the IMS RESLIB.

SYSPRINT DD
This statement must be provided. The data set contains reports of the Advanced ACBGEN utility.

The record format is fixed-blocked, and the logical record length is 121. The block size, if coded, must be a multiple of 121.

Common DD statements

The following DD statements are common to the Advanced ACBGEN utility, IMS ACBGEN utility, and the ACB Generation and Catalog Populate utility (DFS3UACB). These DD statements work the same in all three utilities.

IMS DD
This statement must be provided. It points to the IMS.PSBLIB and IMS.DBDLIB data sets.

IMSACB DD
This statement must be provided. The data set name specified in this DD statement must be the name of the ACB library. Do not use Linkage Editor to place the members in the data set.

SYSIN DD
This statement must be provided. The data set is used for specifying the input control statements.

The record format is fixed-blocked, and the logical record length is 80. The block size, if coded, must be a multiple of 80. During execution, this utility can process as many control statements as required.

For more information about SYSIN DD and SYSIN control statements, see the topic "Application Control Blocks Maintenance utility" in IMS System Utilities.

COMPCRTL DD
This statement must be provided if either PRECOMP or POSTCOMP is specified on the EXEC statement. This statement contains the control input data set to be used by IEBCOPY.

If both PRECOMP and POSTCOMP are specified on the EXEC statement parameters, this data set must be capable of being closed with a reread option.
This data set must contain the following control statement of the form:

```
COPY INDD=IMSACB,OUTDD=IMSACB
```

**SYSUT3 DD**

This statement must be provided if either PRECOMP or POSTCOMP is specified on the EXEC statement. This statement points to a work data set.

**SYSUT4 DD**

This statement works the same as SYSUT3.

**ACBCATWK DD**

This statement is optional. The Advanced ACBGEN utility generates a list of the ACB members that are written to the ACB library during ACB generation. This DD statement is the same as the ACBCATWK DD statement that is used for the IMS ACBGEN utility.

The output written to this data set by the Advanced ACBGEN utility is identical the output written by the IMS ACBGEN utility.

**DD statements used in ACB Generation and Catalog Populate utility (DFS3UACB) jobs**

Use the following DD statements when you generate ACB members with the Advanced ACBGEN utility in ACB Generation and Catalog Populate utility (DFS3UACB) jobs. For a complete list of DD statements, see the topic "ACB Generation and Catalog Populate utility (DFS3UACB)" in *IMS System Utilities*.

**DFS3PPRM DD**

This statement is optional. The DFS3PPRM DD statement specifies execution parameters for the IMS Catalog Populate utility (DFS3PU00).

Use this DD statement to override the default parameters of the DFS3PU00 utility, which is automatically executed after the ACB members are generated. With the default parameters, the DFS3PU00 utility runs in update mode with DBRC and without IRLM.

**IMSACB01 DD**

This statement is optional. This DD statement specifies the ACB library data set that contains the ACB members that are used by the DFS3PU00 utility to populate the IMS catalog.

This DD statement must specify the same data set defined in the IMSACB DD statement. To ensure that the same data set is referenced, code this DD statement with an asterisk as the high-level qualifier, as follows:

```
//IMSACB01 DD DSN=*.ACBLIB,DISP=OLD
```

**PROCLIB DD**

A PROCLIB DD statement must be provided. The PROCLIB DD statement specifies the IMS.PROCLIB data set that contains the DFSDFxxx member that defines various attributes of the IMS catalog that are required by the DFS3PU00 utility to populate the IMS catalog.

**Unique DD statements for the Advanced ACBGEN utility**

The following DD statements are used only by the Advanced ACBGEN utility.

**ACBSYSIN DD**

This statement is optional. The data set can be used to specify parameters used by the Advanced ACBGEN utility. If used, it must contain 80-character, fixed-length records.
Related reading: For the format of the control statements, see “ACBSYSIN control statements” on page 359.

DFSPRINT DD
This statement is optional. The data set, if provided, contains all of the DFSnnnn messages that the SYSPRINT data set would normally contain. Because the Advanced ACBGEN utility places all of its reports in the SYSPRINT data set, the DFSPRINT DD statement can be used to separate the DFSnnnn messages from the reports.

The reports generated by the Advanced ACBGEN utility contain all of the information found in the DFSnnnn messages that were issued during the process of generating ACB members and that would normally be written to the SYSPRINT data set. To isolate and suppress the printing of these verbose and voluminous DFS messages, specify a DFSPRINT DD DUMMY JCL statement.
Control statements for the Advanced ACBGEN utility

The IMS ACBGEN utility is controlled by control statements. There are two control data sets; the SYSIN data set (required) and the ACBSYSIN data set (optional).

The SYSIN data set contains the BUILD and DELETE statements that are required by the ACBGEN process, and the ACBSYSIN data set contains control statements that specify miscellaneous runtime parameters used by the Advanced ACBGEN utility.

These control statements have a fixed length of 80 characters. The control data set can be blocked or unblocked.

**SYSIN control statements**

One or more control statements must be specified in the SYSIN DD statement.

For a full description of these control statements, see the *IMS System Utilities*. The BUILD and DELETE statements in this control data set are processed in exactly the same way as the IMS ACBGEN utility would process them. However, the error messages generated during the parsing of these control statements might differ slightly.

**Syntax rules**

The syntax rules can be summarized as follows:

- Control statement records must be 80 bytes in length.
- Positions 1 - 71 are used, position 72 is used for continuation, and positions 73 - 80 are ignored.
- A statement consists of a label, an operation field, one or more operand fields, and a comment.
- The label is optional. If specified, it must start in position 1.
- The operation field is required and must be preceded and followed by one or more blanks.
- An operand field is required and must follow the operation field. It must be preceded and followed by one or more blanks.
- A comment can be written following the last operand. It must be separated from the operand by one or more blanks.
- Commas, parenthesis, equal signs, and blanks can be used only as delimiting characters.
- If a control statement does not fit within an 80-byte record, it can be continued from one 80-byte record to the next. A continuation is marked by placing a non-blank character in position 72 of the record being continued. The next 80-byte record in the control statement data set must be blank in positions 1 - 15, and the continued text must start in position 16.

**ACBSYSIN control statements**

A single control statement can be specified in the ACBSYSIN DD statement. The syntax rules are different from the syntax rules for the SYSIN control statements. If this DD statement is not present, the default values for the ACBGEN commands are used.

Subsections:
ACBSYSIN control statement example

The following figure shows an example of the ACBSYSIN control statement.

```
ACBGEN  MONITOR=(PROGRESS=(YES,250))
ACBGEN  TYPERUN=PREVUE
ACBGEN  REPORTS=(SYSINLST=NO,PSBLIST=YES)
```

Figure 139. Examples of the ACBSYSIN control statement for the ACBGEN utility

Syntax rules

The control statements for ACBSYSIN must adhere to the following syntax rules:

Control statement content

- The statement text is contained within the first 72 positions of an 80-byte record. The last 8 bytes are ignored.
- A statement consists of the following fields:
  - Label (optional)
  - Command code (required)
  - Operands (optional)
  - Comment (optional)

For example:

```
LABEL  COMMAND  OPERAND=TEST  /* COMMENT */
```

- A complete statement can be as long as several 80-byte records.
- A control statement data set can contain any number of control statements.

Special characters

The following characters have special meaning within a control statement:

- Blank
- Comma
- Equal sign
- Parenthesis
- Single quotation mark
- Decimal point

Continuation characters

- Continuation characters must be used if a control statement does not fit within a single input record.
- A continuation character is either a plus (+) or a minus (-) sign.
- A continuation character must be the last character in the input record.

Literals

- A literal consists of one or more characters enclosed in single quotation marks.
- A single quotation mark within a literal must be represented by two consecutive quotation marks, as in `"ISN"'T`. 
• Literals cannot be continued from one record to the next.

Labels
• A control statement can, optionally, have a label field.
• The label must start in the first position of the control statement.
• The label must consist of a period (.) and 1 - 7 alphanumeric characters, as in .LABEL3.
• The label must be followed by one or more blanks.

Command codes
• A command code consists of predefined words.
• A command code must follow a label (if one is present) and must precede the operands (if any are present).
• A command code must be followed by one or more blanks.

Keyword operands
• A keyword operand consists of a keyword immediately followed by an equal sign. The equal sign can be followed by either a suboperand or one or more optional data values. In REPORTS=(PSBLIST=YES), for example, PSBLIST= is a suboperand of the REPORTS= keyword operand, and YES is the data value for the PSBLIST= keyword operand.
• Keyword operands must be separated by a comma.
• Keyword operand data values can be enclosed in parentheses or quotation marks. For example:
  
  TYPERUN=PREVUE
  TYPERUN=(PREVUE)
  TYPERUN='PREVUE'

• If a keyword operand has suboperands, they must be enclosed in parentheses. For example:
  
  REPORTS=(PSBLIST=YES)
  REPORTS=(PSBLIST=YES,SYSINLST=NO)
  MONITOR=(PROGRESS=(YES,500))

Comments
• A comment must be enclosed within a /* */ pair, such as:
  
  /* THIS IS A COMMENT */

• A comment can appear only at the end of a control statement.

ACBGEN command

An ACBGEN command can be provided to specify the parameters that the Advanced ACBGEN utility is to use during the ACBGEN process. The ACBGEN command can contain the following operands:

PAGESIZE=
  A one- to three-digit number that specifies the number of lines to a page for reports. The default is 60.

REPORTS=
  Valid suboperands are SYSINLST=, PSBLIST=, and LOADSTAT=. Each of these suboperands must specify YES or NO.

SYSINLST=
  The SYSINLST= suboperand controls whether the BUILD and DELETE lists in the SYSIN DD data set are to be listed. If the TYPERUN=PREVUE operand is specified, this suboperand is
This report might be of interest if the SYSIN data set is being generated through some automated procedure and you want to verify its content. The default is SYSINLST=YES.

**PSBLIST=**
The PSBLIST= suboperand controls whether you want a listing of the final list of PSBs generated. If the TYPERUN=PREVUE operand is specified, this suboperand is ignored. This list is constructed from the BUILD and DELETE lists in the SYSIN DD data set. This report might be of interest if you want to see the PSBs that were implicitly included because they refer to one of the DBDs in the BUILD list. The default is PSBLIST=NO.

**LOADSTAT=**
The LOADSTAT= suboperand controls whether you want to display the Load Module Mgmt Stats report. This report is included primarily to give the user some indication of why the ACBGEN process takes as long as it does. The default is LOADSTAT=NO.

**TYPERUN= PREVUE**
The preview feature enables you to preview the list of PSB names that will be generated from your SYSIN control statement data set. Its use terminates the Advanced ACBGEN utility program after all SYSIN control statements have been processed, but before the block building begins. It produces the following reports:

- DBD/PSB Names Specified Via SYSIN report
- ACB Library Members Deleted Due to User Request report
- Final PSB Build List report

This feature is useful, when a DBD is changed in the DBD library and an ACBGEN is required. A BUILD DBD=(dbdname) in the SYSIN data set causes the ACBGEN process to scan the entire ACB library for PSBs that contain references to the specified DBD. Each PSB found is added to the build list. Unless you are familiar with the DBD and its use, you might not know whether there are 10, 100, or 1000 PSBs sensitive to that DBD. Without this information, you do not know whether the ACBGEN will take 1 minute or 1 hour. The PREVUE feature enables you to see the scale of the ACBGEN before scheduling it.

If only the TYPERUN=PREVUE operand is removed, the same JCL and SYSIN data set are used to perform the actual ACBGEN.

**MONITOR=**
This operand enables you to monitor the progress of the ACBGEN process. This option is especially useful during long-running ACBGEN jobs. Progress is measured by the number of PSBs processed. The valid suboperand is PROGRESS=.

**PROGRESS=**
The PROGRESS= suboperand specifies YES or NO and how often you want to be notified. For example, MONITOR=(PROGRESS=(YES,100)) would cause a notification message to be issued each time 100 PSBs are processed. The notification message is time-stamped and sent to the MVS console and the JES job listing. The default is PROGRESS=NO. The frequency value can be blank or a one- to seven-digit number. If PROGRESS=(YES) is specified but the frequency is not, the frequency default is 100.
Output from the Advanced ACBGEN utility

The Advanced ACBGEN utility generates a number of reports, some of which are optional. These reports provide all of the information provided by the utility. The information is in a more concise and organized format. Also, considerable information not previously available is generated.

The same set of reports is generated when the Advanced ACBGEN utility is used in the ACB Generation and Catalog Populate utility (DFS3UACB) job.

SYSPRINT data set

The SYSPRINT data set contains the reports generated by the Advanced ACBGEN utility.

Input Specifications report

This report identifies information to be used as input to this ACB generation execution. Each input parameter might affect the execution in some manner.

Subsections:
- “Sample report”
- “Report field descriptions” on page 364

Sample report

The following figure shows an example of the Input Specifications report.
Figure 140. Input Specifications report (Advanced ACBGEN utility)

Report field descriptions

CONTENTS OF "ACBSYSIN" CONTROL STATEMENT DATASET
This section of the report lists the control statements found in the ACBSYSIN DD statement data set. The ACBSYSIN DD statement is optional. If none is provided, this subreport will not be present.

CONTENTS OF "SYSIN" CONTROL STATEMENT DATASET
This section of the report lists the control statements found in the SYSIN DD statement data set. The SYSIN DD statement is required. It consists of one or more BUILD control statements, DELETE control statements,
both. The DBD and PSB names specified here are listed in the optional DBD/PSB Names Specified via SYSIN report.

CONTENTS OF EXEC STATEMENT PARM FIELD
This section of the report simply lists the content of the PARM field in an EXEC JCL statement.

RUN-TIME PARAMETERS
This section of the report simply lists the parameters used by the Advanced ACBGEN utility. If any parameter was not specified, the default value is shown in this report. A description of each line follows:

IMS RELEASE LEVEL BEING USED
This value is obtained from the DFSVC000 member in your installation’s IMS RESLIB. It is placed in the ACBLIB directory entry for each PSB and DBD added or replaced.

TIMESTAMP USED
This value is placed in the ACBLIB directory entry for each PSB and DBD added or replaced.

BUILD PSB=ALL SPECIFIED
If a BUILD PSB=ALL control statement was found in the SYSIN control data set, this value is set to Y. If this control statement is found, all other SYSIN control statements (if any) are ignored.

PRE-COMPRESSION
This parameter is specified in the PARM field of the EXEC JCL statement. If it is Y, the IEBCOPY utility is invoked to compress the ACB library after all DELETE requests have been processed but before the block building begins. However, if Build PSB=All is specified, pre-compression is not done.

POST-COMPRESSION
This parameter is specified in the PARM field of the EXEC JCL statement. If it is Y, the IEBCOPY utility is invoked to compress the ACB library after the block building process ends.

OPTIONAL REPORTS REQUESTED
The values shown here are the REPORTS=(SYSINLST=) and the REPORTS=(PSBLIST=) operand of the ACBGEN command in the ACBSYSIN control data set. Whether a DFSPRINT DD statement was provided is also shown.

PROGRESS MONITOR REQUESTED
This parameter specifies whether notification of the PSB build progress should be sent to the MVS console.

ACB/PSB/DBD Library Information report
This report shows the dsname of the ACB, PSB, and DBD libraries that were used during the ACBGEN process. The directory information for each library is also shown. For the ACB library, this information reflects the status of the library before processing began. A subsequent report reflects its content after the processing is completed.

The following figure shows an example of the ACB/PSB/DBD Library Information report.
DBD/PSB Names Specified via SYSIN report

This report contains a list of all the names (DBD or PSB) in the BUILD and DELETE statements in the SYSIN data set. This report is useful to view and verify the contents of your SYSIN data set, particularly if you are using some automated process to generate this list.

To generate this report, specify the REPORTS=(SYSINLST=YES) operand of the ACBGEN command in the ACBSYSIN data set.

An example of this report is shown in the following figure.
ACB Library Members Deleted Due to User Request report

This report contains a list of all the DBDs and PSBs that were deleted from ACBLIB before the block building began. Each name listed also appears in a DFS0938 or a DFS0586 message.

To generate this report, request it by use of the REPORTS=(SYSINLIST=YES) operand of the ACBGEN command in the ACBSYSIN data set.

PSB names that are not specified in a DELETE control statement in the SYSIN data set might appear in this list. Deleting a DBD by using DELETE also causes all PSBs that refer to the DBD to be deleted.

An example of this optional report is shown in the following figure.

---

Final PSB Build List report

This report contains the list of PSB names that are passed to Block Builder. Message DFS0587 or DFS0940 is subsequently issued for each name in this list.

To generate this report, request it by use of the REPORTS=(PSBLIST=YES) operand of the ACBGEN command in the ACBSYSIN data set.

A PSB name added implicitly to the list is preceded by an asterisk (*). Any PSB containing a reference to a DBD found in the input build list is said to be implicit.

The total number of the listed PSBs and DBDs are shown at bottom of this report. The DBDs counted are the dbdnames that are specified by a control statement in the SYSIN data set as BUILD DBD=(dbname) and those dbdnames that are found in the ACB library. If a dbdname is not found in the ACB library, IMS message DFS0586 is issued, and it is not included in the count.

The following figure shows an example of this report.
PSB Size Summary report
This report consolidates all the information provided in the DFS0940, DFS0941, DFS0589, and DFS0593 messages into one print line per PSB. The PSBs listed in the report are those added to the ACB library or replaced there.

Subsections:
- “Sample report”
- “Report field descriptions”

Sample report
The following figure contains an example of the PSB Size Summary report.

```
PSB@001  PSB@002  *PSB@007

NUMBER OF PSB = 3  NUMBER OF DBD = 1
NOTE: AN ASTERISK (*) INDICATES NAME WAS INCLUDED IMPLICITLY
```

Figure 144. Final PSB Build List report

### PSB Size Summary report
This report consolidates all the information provided in the DFS0940, DFS0941, DFS0589, and DFS0593 messages into one print line per PSB. The PSBs listed in the report are those added to the ACB library or replaced there.

Subsections:
- “Sample report”
- “Report field descriptions”

Sample report
The following figure contains an example of the PSB Size Summary report.

```
<table>
<thead>
<tr>
<th>PSBNAME</th>
<th>TOTAL SIZE</th>
<th>PSB SIZE</th>
<th>PCB SIZE</th>
<th>WORK AREA</th>
<th>NDX</th>
<th>XIO</th>
<th>IOA</th>
<th>SEG</th>
<th>SSA</th>
<th>SIZE</th>
<th>SAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB0001</td>
<td>14528</td>
<td>12416</td>
<td>3040</td>
<td>2112</td>
<td>448</td>
<td>272</td>
<td>1000</td>
<td>152</td>
<td>200</td>
<td>2880</td>
<td>9280</td>
</tr>
<tr>
<td>PSB0002</td>
<td>7616</td>
<td>6016</td>
<td>1032</td>
<td>1600</td>
<td>264</td>
<td>272</td>
<td>600</td>
<td>152</td>
<td>280</td>
<td>1152</td>
<td>4736</td>
</tr>
<tr>
<td>PSB0007</td>
<td>1344</td>
<td>640</td>
<td>464</td>
<td>704</td>
<td>56</td>
<td>8</td>
<td>600</td>
<td>8</td>
<td>0</td>
<td>576</td>
<td>0</td>
</tr>
</tbody>
</table>

MAX: 14528  12416  3040  2112  448  272  1000  152  200  2880  9280
AVG: 1536  4672

```

Figure 145. PSB Size Summary report

Report field descriptions
The last two lines in the report, labeled MAX: and AVG: , contain the information from the DFS0591 and the DFS0942 messages. For the meaning of each column in this report, see IMS Messages and Codes.
**PSBNAME**
The PSB names shown here are displayed in DFS messages DFS0589, DFS0593, DFS0940, and DFS0941.

**TOTAL SIZE**
This data was extracted from message DFS0589.

**PSB SIZE**
This data was extracted from message DFS0589. It also appears in message DFS0940.

**PCB SIZE**
This data was extracted from message DFS0589.

**WORK AREA**
This data was extracted from message DFS0589.

**WORKAREA BREAKOUT -- NDX**
This data was extracted from message DFS0593.

**WORKAREA BREAKOUT -- XIO**
This data was extracted from message DFS0593.

**WORKAREA BREAKOUT -- IOA**
This data was extracted from message DFS0593.

**WORKAREA BREAKOUT -- SEG**
This data was extracted from message DFS0593.

**WORKAREA BREAKOUT -- SSA**
This data was extracted from message DFS0593.

**CSA SIZE**
This data was extracted from message DFS0941.

**SAS SIZE**
This data was extracted from message DFS0941.

**MAX**
This data was extracted from messages DFS0591 and DFS0942.

**AVERAGE**
This data was extracted from message DFS0942.

**PSB/DBD Change Summary report**
This report consolidates all the information provided in the DFS0940 and DFS0960 messages into one print line per DBD or PSB, plus provide some information about those DBDs and PSBs. All DBDs are listed first followed by all PSBs in the ascending name sequence.

Subsections:
- "Sample report"
- "Report field descriptions" on page 370

**Sample report**
The following figure shows an example of the PSB/DBD Change Summary report.
### Report field descriptions

**DBD/PSB NAME**  
Self-explanatory.

**TYPE**  
Self-explanatory.

**ACTION TAKEN**  
The action taken for a DBD. This can be ADDED, REPLACED, NOT REPL, or DELETED.

The ADDED designation is generated if the DBD does not exist in ACBLIB at block building time and is reflected in a DFS0940 message. If the user has requested by issuing a DELETE DBD control statement, that the DBD be deleted from ACBLIB before block building begins, the DFS0940 message describes it as ADDED. However, the Advanced ACBGEN utility

---

**TABLE: PSB/DBD Change Summary**

<table>
<thead>
<tr>
<th>DBD/PSB NAME</th>
<th>TYPE</th>
<th>ACTION</th>
<th>OLD SIZE</th>
<th>NEW SIZE</th>
<th>OLD GEN DATE</th>
<th>NEW GEN DATE</th>
<th>#DMBS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB0001</td>
<td>PSB</td>
<td>ADDED</td>
<td>12416</td>
<td></td>
<td>10/01/15</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>PSB0002</td>
<td>PSB</td>
<td>ADDED</td>
<td>6016</td>
<td></td>
<td>10/01/15</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>PSB0006</td>
<td>PSB</td>
<td>DELETED</td>
<td>512</td>
<td></td>
<td>10/01/15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSB0007</td>
<td>PSB</td>
<td>REPLACED</td>
<td>640</td>
<td>640</td>
<td>10/01/15</td>
<td>10/01/15</td>
<td>1</td>
</tr>
</tbody>
</table>

**TABLE: DBD Change Summary**

<table>
<thead>
<tr>
<th>DBD/PSB NAME</th>
<th>TYPE</th>
<th>ACTION</th>
<th>OLD SIZE</th>
<th>NEW SIZE</th>
<th>OLD GEN DATE</th>
<th>NEW GEN DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD0001A</td>
<td>DBD</td>
<td>ADDED</td>
<td>704</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0003A</td>
<td>DBD</td>
<td>ADDED</td>
<td>1280</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0001A</td>
<td>DBD</td>
<td>ADDED</td>
<td>1408</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0002A</td>
<td>DBD</td>
<td>ADDED</td>
<td>2560</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0001A</td>
<td>DBD</td>
<td>ADDED</td>
<td>640</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0002A</td>
<td>DBD</td>
<td>ADDED</td>
<td>1472</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0101A</td>
<td>DBD</td>
<td>ADDED</td>
<td>640</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0101A</td>
<td>DBD</td>
<td>ADDED</td>
<td>768</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0102A</td>
<td>DBD</td>
<td>ADDED</td>
<td>448</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0103A</td>
<td>DBD</td>
<td>ADDED</td>
<td>960</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0002A</td>
<td>DBD</td>
<td>ADDED</td>
<td>256</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0003A</td>
<td>DBD</td>
<td>DELETED</td>
<td>192</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0004A</td>
<td>DBD</td>
<td>REPLACED</td>
<td>192</td>
<td>10/01/15</td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0002A</td>
<td>DBD</td>
<td>ADDED</td>
<td>704</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0001A</td>
<td>DBD</td>
<td>ADDED</td>
<td>640</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0002A</td>
<td>DBD</td>
<td>ADDED</td>
<td>448</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0003A</td>
<td>DBD</td>
<td>ADDED</td>
<td>448</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0004A</td>
<td>DBD</td>
<td>ADDED</td>
<td>448</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0005A</td>
<td>DBD</td>
<td>ADDED</td>
<td>448</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0006A</td>
<td>DBD</td>
<td>ADDED</td>
<td>640</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0007A</td>
<td>DBD</td>
<td>ADDED</td>
<td>448</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
<tr>
<td>DBD0008A</td>
<td>DBD</td>
<td>ADDED</td>
<td>448</td>
<td></td>
<td>10/01/15</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** This report reflects the action taken in the following DFS messages: 0587, 0938, 0940, and 0960.

---

*Figure 146. PSB/DBD Change Summary report*

**Report field descriptions**

**DBD/PSB NAME**  
Self-explanatory.

**TYPE**  
Self-explanatory.

**ACTION TAKEN**  
The action taken for a DBD. This can be ADDED, REPLACED, NOT REPL, or DELETED.

The ADDED designation is generated if the DBD does not exist in ACBLIB at block building time and is reflected in a DFS0940 message. If the user has requested by issuing a DELETE DBD control statement, that the DBD be deleted from ACBLIB before block building begins, the DFS0940 message describes it as ADDED. However, the Advanced ACBGEN utility
reports this DBD as having been REPLACED, because logically that is what happened. Thus, the only DBDs that are shown as ADDED are those that do not appear in any DELETE or BUILD control statements, or that do not appear in ACBLIB.

The NOT REPL DBDs are those that already exist in ACBLIB and were not specified in a BUILD DBD control statement.

The REPLACED DBDs are those that were specified in a BUILD DBD control statement and subsequently referred to by one of the PSBs generated.

The DELETED DBDs are those that were deleted by means of a DELETE DBD control statement and have not been replaced.

If a BUILD PSB=ALL control statement was specified, the action taken can be ADDED, DELETED, or REPLACED. The old ACBLIB is used as the basis for setting the action taken.

**SIZE**

The size shown is the size of the member written into the ACB library. The value shown appears in the ACB directory entry for that DBD or PSB. However, be aware that this value is rounded down before it is placed in the directory entry, because the directory entry uses a 2-byte field. PSB sizes shown in this report are divided by 16 and DBDs by 8 before being placed into a directory entry.

**OLD GEN DATE**

This field shows the previous ACBGEN date for the DBD or PSB. This date is extracted from the ACBLIB directory entry before the block building process begins. Any DBD or PSB marked as ADDED obviously does not have the date on which the DBD was generated.

**NEW GEN DATE**

This field shows the ACBGEN date that is now stored in the ACBLIB for a DBD or PSB that has been ADDED or REPLACED in the ACB library.

**#DMBs REFERENCED**

This column applies only to PSBs. It is a count of the number of unique DBD names referred to by this PSB.

**Miscellaneous DFS Messages report**

This report shows all the DFS-type messages generated that are not accounted for by the PSB Size Summary and PSB/DBD Change Summary reports. Any DFSnnnn messages which also set a nonzero return code are in this list.

The following figure shows an example of the Miscellaneous DFS Messages report.
DFS Messages Summary report

This report consolidates the numerous DFS-type messages generated into a single, concise report.

The report contains a line for each unique DFS-type message number generated. The print line identifies the DFS message number, the quantity of those messages, and the message text. The message text has been paraphrased for ease of reading (and also to make it fit in the print line). The specific DBD or PSB is shown in the DFS Messages Detail report. Some of the DFS message number repeats, that is, two print lines with the same DFS message number. One of the two lines shows DBD counts and the other shows PSB counts.

If PSB build fails with message DFS0649W, which indicates that the storage is insufficient, Advanced ACBGEN resolves the storage shortage problem, issues an FABQ9993I message, and starts rebuilding the PSB. With the IMS ACBGEN utility (DFSUACB0), the PSB is deleted from ACBLIB when a PSB build fails with a DFS0649W message, but with Advanced ACBGEN, the PSB is not deleted from ACBLIB. A DFS0940I message is issued when PSB build succeeds.

The following figure shows an example of the DFS Messages Summary report.

<table>
<thead>
<tr>
<th>MESSAGE ID</th>
<th>MESSAGE TEXT</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFS0586I</td>
<td>DBD 'DTA1' REQUESTED IN DELETE OPERATION NOT FOUND IN ACBLIB - REQ</td>
</tr>
<tr>
<td>DFS0586I</td>
<td>DBD 'DI21XXXX' REQUESTED IN BUILD OPERATION NOT FOUND IN ACBLIB - REQ</td>
</tr>
<tr>
<td>DFS0586I</td>
<td>DBD 'DTA1' REQUESTED IN BUILD OPERATION NOT FOUND IN ACBLIB - REQ</td>
</tr>
<tr>
<td>DFS0929I</td>
<td>BLDL FAILED FOR MEMBER --PSB1</td>
</tr>
<tr>
<td>DFS0587I</td>
<td>ERROR BUILDING PSB=PSB1 - IT WILL BE DELETED FROM ACBLIB. *****</td>
</tr>
<tr>
<td>DFS0929I</td>
<td>BLDL FAILED FOR MEMBER --PSB2</td>
</tr>
<tr>
<td>DFS0587I</td>
<td>ERROR BUILDING PSB=PSB2 - IT WILL BE DELETED FROM ACBLIB. *****</td>
</tr>
<tr>
<td>DFS0929I</td>
<td>BLDL FAILED FOR MEMBER --PSB3</td>
</tr>
<tr>
<td>DFS0587I</td>
<td>ERROR BUILDING PSB=PSB3 - IT WILL BE DELETED FROM ACBLIB. *****</td>
</tr>
<tr>
<td>DFS0962I</td>
<td>DBD DI21XXXX NOT PROCESSED. NO ACBLIB PSB REFERENCES THE NAMED DBD.</td>
</tr>
<tr>
<td>DFS0590I</td>
<td>END OF ACBLIB MAINTENANCE. HIGHEST CONDITION CODE WAS 00000016</td>
</tr>
</tbody>
</table>

Figure 147. Miscellaneous DFS Messages report
Run Summary report

This report provides a concise summary of the ACB/PSB/DBD library activity, and also shows the completion code that is passed back to the MVS job step termination routine.

Subsections:

- **“Sample report”**
- **“Report field descriptions”** on page 374

Sample report

The following figure shows an example of the Run Summary report.

---

**Figure 148. DFS Messages Summary report**

<table>
<thead>
<tr>
<th>MESSAGE ID</th>
<th>QTY</th>
<th>MESSAGE TEXT (PARAPHRASED)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFS0589</td>
<td>3</td>
<td>PROCESSING COMPLETED FOR PSB</td>
</tr>
<tr>
<td>DFS0590</td>
<td>1</td>
<td>END OF ACBLIB MAINTENANCE</td>
</tr>
<tr>
<td>DFS0591</td>
<td>1</td>
<td>MAX SIZES OF PSB COMPONENTS</td>
</tr>
<tr>
<td>DFS0593</td>
<td>3</td>
<td>PSB WORKAREA BREAKOUT</td>
</tr>
<tr>
<td>DFS0938</td>
<td>1</td>
<td>DELETE PROCESSING COMPLETED FOR DBD</td>
</tr>
<tr>
<td>DFS0938</td>
<td>1</td>
<td>DELETE PROCESSING COMPLETED FOR PSB</td>
</tr>
<tr>
<td>DFS0940</td>
<td>21</td>
<td>DBD HAS BEEN ADDED/REPLACED IN ACBLIB</td>
</tr>
<tr>
<td>DFS0940</td>
<td>3</td>
<td>PSB HAS BEEN ADDED/REPLACED IN ACBLIB</td>
</tr>
<tr>
<td>DFS0941</td>
<td>3</td>
<td>PSB CSA AND SAS SIZES IF USING DL/I SEPARATE ADDR SPACE</td>
</tr>
<tr>
<td>DFS0942</td>
<td>1</td>
<td>MAX AND AVERAGE CSA AND SAS SIZES IF USING SEPARATE ADDR SPACE</td>
</tr>
</tbody>
</table>
Report field descriptions

NUMBER OF BUILD PSB FAILURES
The number of PSBs that were not added to or replaced in the ACB library because of some error condition. If the completion code is 8, this line indicates how many errors occurred. The PSB name and the error condition are given in the Miscellaneous DFS Messages report.

NUMBER OF PSBS ADDED TO ACBLIB
The number of PSBs that were added to the ACB library. The PSBs counted here are those that did not exist in the ACB library before this execution of the Advanced ACBGEN utility.

NUMBER OF PSBS DELETED FROM ACBLIB
The number of PSBs that were deleted from the ACB library and not replaced. The deletion could have been initiated explicitly from a DELETE PSB= control statement in the SYSIN data set, or implicitly because of some error during the building process. A DFS message is always issued for each PSB deleted.

NUMBER OF PSBS REPLACED IN ACBLIB
The number of PSBs that were replaced in the ACB library. Those PSBs which were explicitly deleted and then added to the ACB library are reflected here.

NUMBER OF DMBS ADDED TO ACBLIB
The number of DMBs that were added to the ACB library. The DMB did not exist in the ACB library before this execution of the Advanced ACBGEN utility.
NUMBER OF DMBS DELETED FROM ACBLIB
The number of DMBs that were deleted from the ACB library. The deletion could have been initiated explicitly from a DELETE DBD= control statement in the SYSIN data set, or implicitly because of some error during the building process. A DFS message is always issued for each DMB deleted.

NUMBER OF DMBS REPLACED IN ACBLIB
The number of DMBs that were replaced in the ACB library. DMBs that were explicitly deleted and then added to the ACB library are reflected here.

NUMBER OF DMBS NOT REPLACED IN ACBLIB
The number of DMBs that were not replaced in the ACB library. These are DMBs that were generated as part of the ACBGEN process but were not explicitly requested to be built with a BUILD DBD= control statement in the SYSIN data set. Each of these DMBs might be referenced more than once in a DFS0960 message.

NUMBER OF LOAD PSBS ISSUED
The number of PSBs that were loaded during the block building process. This number should be equal to the number of PSBs generated.

NUMBER OF LOAD DBDS ISSUED
The number of DBDs that were loaded during the block building process, which is the number of DBDs referred to by the PSBs being generated. This number is presented for information only; however, it obviously has an effect on how long the ACBGEN process takes.

NUMBER OF ACBLIB MEMBERS
The number of members (directory entries) that are in the ACB library at the completion of the ACBGEN process. The ACB/PSB/DBD Library Information subreport shows this information before the ACBGEN process.

NUMBER OF ACBLIB DIRECTORY BLKS USED
The number of ACB library directory blocks that were used at the completion of the ACBGEN process. The ACB/PSB/DBD Library Information subreport shows this information before the ACBGEN process.

NUMBER OF ACBLIB DIRECTORY BLKS UNUSED
The number of ACB library directory blocks that were allocated but not used at the completion of the ACBGEN process. The ACB/PSB/DBD Library Information subreport shows this information before the ACBGEN process. Because each directory block can hold six directory entries, this should enable you to calculate how many more DMB/PSBs can be added to the library.

START TIME
The time at which the ACBGEN process began. This time stamp is used in all ACB library directory entries that were replaced or added.

END TIME
The time at which the ACBGEN process ended.

COMPLETION CODE
The job step condition code. It is passed back to the MVS job step termination routine and is available for COND= testing in subsequent job steps within the same job. If the value is a nonzero value, a DFS message is issued to notify you of the warning or error condition. The highest return code encountered is the one reported here.
Load Module Management Status (Load Module Mgmt Stats) report

To reduce the elapsed time, the Advanced ACBGEN utility caches the loaded DBD members in the internal work storage (DBD hold area) and reduces I/O operations. This report provides statistics about the DBD hold area.

This report is printed more than once when PSB build is retried.

Subsections:
- “Sample report”
- “Report field descriptions”

Sample report

The following figure shows an example of the Load Module Management Status report.

```
+----------------------+
|LOAD MODULE MGMT STATS|
+----------------------+
MAX ENTRIES FOR TOC=47
TOC SIZE=940
DBD HOLD AREA SIZE REQUESTED=48128
DBD HOLD AREA SIZE OBTAINED=48128
LOCATION OF HOLD AREA=ABOVE 16M
SIZE OF LOAD MOD INPUT AREA=90000
TOC ENTRIES USED=21
DBD HOLD AREA USED=10416
NBR DBDS IN HOLD AREA=21
NBR OF DBD LOAD REQUESTS=25
NBR LOAD REQS FOUND IN HOLD=4
NBR OF DBD DELETE REQUESTS=25
SIZE OF LARGEST USED LOAD MOD=1864
```

Figure 150. Load Module Management Status (Load Module Mgmt Stats) report

Report field descriptions

MAX ENTRIES FOR TOC

TOC stands for Table of Contents. The value shown in this field is the number of entries that are kept in the DBD hold area. This number is based on the DBD members found in the IMS DD statement concatenation.

TOC SIZE

The size of the table of contents area. This area manages the DBD hold area.

DBD HOLD AREA SIZE REQUESTED

The size of the DBD hold area that is requested internally.

DBD HOLD AREA SIZE OBTAINED

This value shows the amount of memory actually obtained for the DBD HOLD Area.
LOCATION HOLD AREA=ABOVE 16M
Currently, all memory obtained by the Library Management routine is above the 16 MB line.

TOC ENTRIES USED
This value is the number of Table of Contents entries that were actually used.

DBD HOLD AREA USED
This value is the actual amount of DBD Hold Area space that was actually used by the DBDs loaded.

NBR DBDS IN HOLD AREA
This value is the actual number of DBDs placed in the Hold Area.

NBR OF DBD LOAD REQUESTS
This value is the actual number of LOAD DBD requests.

NBR LOAD REQS FOUND IN HOLD
This value is the actual number of LOAD DBD requests. That is, this value is the number of LOAD macro requests that were avoided.

NBR OF DBD DELETE REQUESTS
This value is the actual number of DELETE DBD requests.

DFSPRINT data set
The DFSPRINT data set, which is an optional data set, contains the DFSnnnn messages.

The following figure shows messages that are generated in the DFSPRINT data set.

---

DFS0940I DBD DBDATA0 HAS BEEN ADDED IN LIBRARY. DMG SIZE = 00000576 BYTES
DFS0940I PSB PSBLIU01 HAS BEEN ADDED IN LIBRARY. PSB SIZE = 00002752 BYTES
DFS0941I PSB PSBLIU01 IF USING DL/I SEPARATE ADDRESS SPACE, CSA SIZE = 00000320, SAS SIZE = 00002368.
DFS0589I PROCESS COMPLETE FOR PSB-PSBLIU01. PCB = 0000296, PSB = 00002752, WORKAREA = 00001728, TOTAL SIZE = 00004480
DFS0593I PSB-PSBLIU01 WORKAREA BREAKOUT. NDX = 0000056, XIO = 0000272, IOA = 0000600, SEG = 000232, SSA = 000560
DFS0941I PSB PSBLIU01 IF USING DL/I SEPARATE ADDRESS SPACE, CSA SIZE = 00000320, SAS SIZE = 00002368. AVERAGE CSA = 00000320 AVERAGE SAS = 00002368.
DFS0591I MAX PCB SIZE = 0000296, MAX PSB SIZE = 00002752, MAX WORKAREA SIZE = 00001728, MAX TOTAL SIZE = 00004480
DFS0942I IF USING DL/I SAS, MAX CSA = 00000320 MAX SAS = 00002368 AVERAGE CSA = 00000320 AVERAGE SAS = 00002368.
DFS0590I END OF ACBLIB MAINTENANCE. HIGHEST CONDITION CODE WAS 00000000

---

Figure 151. Messages in the DFSPRINT data set

MVS console and the JES job listing
When you request to display the progress of the ACBGEN process with the MONITOR=(PROGRESS=(YES,value)) option in the ACBSYSIN data set, notification messages are displayed on the MVS console and the JES job listing.

The following figure shows an example of the notification messages that are displayed on the MVS console and the JES job listing when MONITOR=(PROGRESS=(YES,1000)) is specified in the ACBSYSIN data set.
Figure 152. Messages in the MVS console and the JES job listing
Chapter 11. ACBLIB Analyzer utility

The ACBLIB Analyzer utility analyzes the ACB libraries and generates several reports that provide detailed information about the ACB libraries.

Topics:
- “ACBLIB Analyzer utility overview” on page 380
- “Analyzing ACB libraries” on page 381
- “JCL requirements for the ACBLIB Analyzer utility” on page 382
- “ACBSYSIN control statements” on page 384
- “Output from the ACBLIB Analyzer utility” on page 386
ACBLIB Analyzer utility overview

The ACBLIB Analyzer utility provides an MVS batch utility program to analyze ACB libraries. It verifies that all ACB library members are at the same IMS version and release level, and that all of them were placed in the ACB library by the ACBGEN process; that is, the library was not inadvertently used during a DBDGEN or PSBGEN. Also, the utility program produces several reports.
Analyzing ACB libraries

To analyze the ACB libraries by using the ACBLIB Analyzer utility, you must prepare the JCL for the ACBLIB Analyzer utility and submit the job.

About this task

Sample JCL for the ACBLIB Analyzer utility is in the SHPSJCL0 library, member FABQIVP. You can modify this sample JCL and then use it to run the utility.

Procedure

1. In the ACBLIB Analyzer JCL, code the EXEC statement and DD statements. See “JCL requirements for the ACBLIB Analyzer utility” on page 382.
2. Optionally, code the control statements for ACBLIB Analyzer in the ACBSYSIN data set. See “ACBSYSIN control statements” on page 384.
3. Submit the job.
4. Check the output data sets that are generated. See “Output from the ACBLIB Analyzer utility” on page 386.
JCL requirements for the ACBLIB Analyzer utility

The ACBLIB Analyzer utility provides a batch utility to analyze ACB libraries. When you code the JCL to run the ACBLIB Analyzer utility, include the EXEC statement and appropriate DD statements.

Subsections:
- “JCL examples”
- “EXEC statement”
- “DD statements”

JCL examples

The following JCL example reports on an ACB library.

```jcl
//ANALYZE EXEC PGM=FABQCHEK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//SYSPRINT DD SYSOUT=* 
//DBTSNAP DD SYSOUT=* 
//IMSACB DD DISP=SHR,DSN=IMSVS.ACBLIB
//ACBSYSIN DD *
   LISTLIB LIBTYPE=ACB,SNAP=(DIRENTRY=N,DIRRCD=N)
/*
```

Figure 153. ACBLIB Analyzer utility JCL example: obtain reports on an ACB library

The following JCL example reports on a DBD library. To use this example to report on a PSB library, replace DBD with PSB. For DBD and PSB libraries, ACBLIB Analyzer prints only two reports: Library Information report and Input Specifications report.

```jcl
//ANALYZE EXEC PGM=FABQCHEK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//SYSPRINT DD SYSOUT=* 
//DBTSNAP DD SYSOUT=* 
//IMSDBD DD DISP=SHR,DSN=IMSVS.DBDBLIB
//ACBSYSIN DD *
   LISTLIB LIBTYPE=DBD,INDO=IMSDBD
/*
```

Figure 154. ACBLIB Analyzer utility JCL example: obtain reports on a DBD library

EXEC statement

The EXEC JCL statement must specify a program name of FABQCHEK. No PARM operand is required.

DD statements

Code the following DD statements to identify the source of input and the placement of output information:

STEPLIB DD
A STEPLIB DD statement (or a JOBLIB DD) must be provided. The data set name specified in this DD statement must be the name of the load
module library containing the ACBLIB Analyzer utility. When the LISTLIB command parameter USESORT=YES in the ACBSYSIN control statement is specified, the suitable sort program library must always be added in a STEPLIB DD (or a JOBLIB DD) statement.

**SYSPRINT DD**
A SYSPRINT DD statement must be provided. This data set contains various reports that are generated by the ACBLIB Analyzer utility.

**Related reading:** For information about the reports generated by the ACBLIB Analyzer utility, see “Output from the ACBLIB Analyzer utility” on page 386.

**FABQRPT DD**
A FABQRPT DD statement can be provided. This output data set is used only when the GENDATE=YES operand is specified in the LISTLIB command in the ACBSYSIN control statement and the library type is ACB (LIBTYPE=ACB). When the GENDATE=YES operand is specified, this data set contains all of the reports except the Input Specifications report.

If the GENDATE=YES operand is specified in the LISTLIB command and the FABQRPT DD statement is not specified, the reports are routed to SYSOUT=*.

The record format is fixed block (FB). The logical record length is 121. If the block size is coded, the block size must be a multiple of 121.

**IMSACB DD**
An IMSACB DD statement must be provided. The DSN= operand must specify the ACB library you want to analyze. The data set is opened for input only, and is not modified by the job.

**ACBSYSIN DD**
An ACBSYSIN DD statement can be provided. The data set is used to specify parameters used by the program. It must contain 80-character, fixed-length records.

**Related reading:** For the content of these records, see “ACBSYSIN control statements” on page 384.

**DBTSNAP DD**
If you specify the SNAP= operand in the ACBSYSIN control statement data set, a DBTSNAP DD statement is required.

**SORT DD**
When the LISTLIB command parameter USESORT=YES in the ACBSYSIN control statement is specified and the user sort program requires some DD statements, the DD statements must be specified in the JCL.
ACBSYSIN control statements

You can specify a single control statement in the ACBSYSIN DD statement. The syntax rules are the same as those of the ACBSYSIN control statements for the Advanced ACBGEN utility program. If the ACBSYSIN DD statement is omitted, the default operand values for the LISTLIB command are used.

Subsections:
- “Control statement example”
- “Syntax rules”
- “LISTLIB command”

Control statement example

The following figures show examples of ACBSYSIN control statements.

LISTLIB
LISTLIB LIBTYPE=ACB,Snap=(DIRENTRY=(Y,100),DIRRCN=N)
LISTLIB LIBTYPE=ACB,USESORT=YES
LISTLIB LIBTYPE=ACB,GENDATE=YES

Figure 155. ACBSYSIN control statement examples for the ACBLIB Analyzer utility: obtain reports on an ACB library

LISTLIB LIBTYPE=DBD,INDD=IMSDBD
LISTLIB LIBTYPE=PSB,INDD=IMSPSB

Figure 156. ACBSYSIN control statement examples for the ACBLIB Analyzer utility: obtain reports on a DBD or PSB library

For DBD and PSB libraries, ACBLIB Analyzer prints only two reports: Input Specifications report and Library Information report.

Syntax rules

The syntax rules for the ACBSYSIN control statement are the same as for the Advanced ACBGEN utility. See “Syntax rules” on page 360 for the syntax rules of the ACBSYSIN control statement.

LISTLIB command

You can provide a LISTLIB command to specify parameters used by ACBLIB Analyzer utility. The LISTLIB command can contain the following operands:

LIBTYPE=
  Identifies the type of the library to process. The operand is DBD, PSB, or ACB. The default is ACB. You can specify only one type of library for one job step.

  If you specify DBD or PSB, only two reports are generated: Input Specifications report and Library Information report.
SNAP=
The valid suboperands for the SNAP operand are DIRENTRY= and DIRRCD=. Each of these suboperands can specify YES or NO, and a number.

The SNAP operand is ignored if the library type is DBD or PSB (LIBTYPE=DBD or LIBTYPE=PSB).

DIRENTRY=
The DIRENTRY= suboperand specifies whether you want a hex dump of any of the library directory entries. If you specify YES, you can also specify a three-digit number which specifies the maximum number of entries to be displayed. The default is DIRENTRY=NO. The requested number of directory entries is displayed (in hexadecimal format) in the Library Contents report.

DIRRCD=
The DIRRCD= suboperand specifies whether you want a listing of any of the library directory records. The default is DIRRCD=NO. If you specify YES, you can also include a three-digit number specifying the maximum number of entries to be displayed. The hex dump is placed in the DBTSNAP DD statement.

USESSORT=
Specifies whether the ACBLIB Analyzer utility is to use the SORT program of your location. The valid operands are YES and NO; the default is USESORT=NO.

If USESORT=YES is specified, the ACBLIB Analyzer utility links to a program named SORT. DFSORT (Data Facility Sort), which is a part of z/OS, or a functionally equivalent sort program is necessary. If the SORT program of your location requires a SYSOUT DD statement, you must specify it in your JCL.

If the ACB library being analyzed contains a large number such as over 1000 PSBs and DMBs, specify USESORT=YES to reduce the job-step elapse time and the CPU utilization time. (This depends on the user environment.)

The USESORT operand is ignored if the type of the library is DBD or PSB (LIBTYPE=DBD or LIBTYPE=PSB).

INDD=
Specifies the ddname of the DD statement that is used as input. The default ddname is IMSACB.

Specify the ddname that defines the data set name of the library to analyze. When an INDD operand is specified, the ACBLIB Analyzer utility analyzes the data set that is defined by the ddname.

GENDATE=
Specifies to include the date and time when the ACBLIB members were generated in the Library Contents report. The value can be YES or NO. The default is GENDATE=NO.

If GENDATE=YES is specified, all of the reports except the Input Specifications report are generated in the FABQRPT data set.

If GENDATE=YES is specified and the LIBTYPE=DBD or LIBTYPE=PSB operand is specified, the GENDATE=YES operand is ignored.
Output from the ACBLIB Analyzer utility

The ACBLIB Analyzer utility generates a number of reports, some of which are optional.

SYSPRINT data set

This data set contains the reports that were generated by the ACBLIB Analyzer utility. However, when GENDATE=YES is specified in the LISTLIB command, all of the reports except the Input Specifications report are generated in the FABQRPT data set instead of the SYSPRINT data set.

FABQRPT data set

When GENDATE=YES is specified in the LISTLIB command, this data set contains the following ACBLIB Analyzer utility reports.

- Library Information report
- Library Contents report
- Distribution of Member Sizes report
- Distribution of PSB Workarea Sizes report
- Chronological History of ACBGENs report
- Warning report

When these reports are generated in the FABQRPT data set, the width of the reports is wider than when they are generated in the SYSPRINT data set. In the FABQRPT data set, the Library Contents report includes date and time fields. The content of all of the reports except the Library Contents report are the same as the reports that are generated when GENDATE=YES is not specified.

Input Specifications report

This report contains the information that was specified as input to this execution of the ACBLIB Analyzer utility.

Subsections:

- "Sample report"
- "Report field descriptions" on page 387

Sample report

The following figure shows an example of the Input Specifications report.
CONTENTS OF "ACBSYSIN" CONTROL STATEMENT DATASET:

This section of the report lists the control statements found in the ACBSYSIN DD statement data set. The ACBSYSIN DD statement is optional. If none is provided, this subreport is not present.

Library Information report

This report shows the library being used and information in the directory of that library.

This report is generated in the SYSPRINT data set. However, when GENDATE=YES is specified on the LISTLIB command, this report is generated in the FABQRPT data set instead of the SYSPRINT data set.

Subsections:

• “Sample report”

• “Report field descriptions” on page 388

Sample report

The following figure shows an example of the Library Information report.

Figure 157. Input Specifications report (ACBLIB Analyzer utility)
**LIBRARIES USED**

This section displays the name and volume serial number of the data set being used. The LIB field is obtained from the LIBTYPE= operand in the LISTLIB command in the ACBSYSIN data set. HIGH-TTR displays the largest DASD TTR found for a member. TTR stands for the relative track and record number used by the data set.

**DIRECTORY INFORMATION**

This section shows information found in the directory where library is located. The column headings are self-explanatory.

**Library Contents report**

This report lists information about the members found in the specified library. All DMBs are shown first followed by all PSBs.

This report is generated in the SYSPRINT data set. However, when GENDATE=YES is specified on the LISTLIB command, this report is generated in the FABQRPT data set instead of the SYSPRINT data set.

This report is generated only for ACB libraries. If the input for the utility is a DBD library or a PSB library (LIBTYPE=DBD or LIBTYPE=PSB), this report is not generated.

**Subsections:**

- “Sample report” on page 389
- “Report field descriptions” on page 391
Sample report

The following figure shows an example of the Library Contents report when GENDATE=YES is not specified in the LISTLIB command.

<table>
<thead>
<tr>
<th>MBRNAME</th>
<th>TYPE</th>
<th>LVL</th>
<th>SIZE (BYTES)</th>
<th># DMBS</th>
<th># PSBS</th>
<th>REFERENCE</th>
<th>REFERENCING</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD0001A</td>
<td>DMB</td>
<td>13.1</td>
<td>648</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0003A</td>
<td>DMB</td>
<td>13.1</td>
<td>1,192</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD00E01A</td>
<td>DMB</td>
<td>13.1</td>
<td>1,336</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD00E02A</td>
<td>DMB</td>
<td>13.1</td>
<td>2,504</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD00H01A</td>
<td>DMB</td>
<td>13.1</td>
<td>600</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD00H02A</td>
<td>DMB</td>
<td>13.1</td>
<td>1,344</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD01S01A</td>
<td>DMB</td>
<td>13.1</td>
<td>560</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD01O01A</td>
<td>DMB</td>
<td>13.1</td>
<td>680</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD01O02A</td>
<td>DMB</td>
<td>13.1</td>
<td>392</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD01O03A</td>
<td>DMB</td>
<td>13.1</td>
<td>872</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0M02A</td>
<td>DMB</td>
<td>13.1</td>
<td>200</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0M04A</td>
<td>DMB</td>
<td>13.1</td>
<td>120</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0S02A</td>
<td>DMB</td>
<td>13.1</td>
<td>616</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0X01A</td>
<td>DMB</td>
<td>13.1</td>
<td>568</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0X02A</td>
<td>DMB</td>
<td>13.1</td>
<td>416</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0X03A</td>
<td>DMB</td>
<td>13.1</td>
<td>416</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0X04A</td>
<td>DMB</td>
<td>13.1</td>
<td>416</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0X05A</td>
<td>DMB</td>
<td>13.1</td>
<td>416</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0X06A</td>
<td>DMB</td>
<td>13.1</td>
<td>568</td>
<td>2</td>
<td>ALIAS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0X07A</td>
<td>DMB</td>
<td>13.1</td>
<td>416</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD0X08A</td>
<td>DMB</td>
<td>13.1</td>
<td>416</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 159. Library Contents report in the SYSPRINT data set (Part 1 of 2)
Figure 160. Library Contents report in the SYSPRINT data set (Part 2 of 2)

The following figure shows an example of the Library Contents report when GENDATE=YES is specified in the LISTLIB command.

<table>
<thead>
<tr>
<th>MBRNAME</th>
<th>TYPE</th>
<th>IMS</th>
<th>SIZE (BYTES)</th>
<th># DMBS</th>
<th># PSBS</th>
<th>REFERENCE</th>
<th>REFERENCING</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB0001</td>
<td>PSB</td>
<td>13.1</td>
<td>10,944</td>
<td>16</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSB0002</td>
<td>PSB</td>
<td>13.1</td>
<td>4,944</td>
<td>8</td>
<td></td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSB0005</td>
<td>PSB</td>
<td>13.1</td>
<td>288</td>
<td>0</td>
<td></td>
<td>0</td>
<td></td>
<td>NO DBDS REFERENCED</td>
</tr>
<tr>
<td>PSB0007</td>
<td>PSB</td>
<td>13.1</td>
<td>512</td>
<td>1</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 161. Library Contents report in the FABQRPT data set (Part 1 of 2)**
The DETAIL section contains the following fields:

**MBRNAME and TYPE**
These columns give the name and type (DMB or PSB) of the member.

**IMS LVL**
This column shows the IMS version and release level that was used to generate this member.

**SIZE**
This column shows the size of the member record in the ACB library. This value appeared in the DFS0940 message during the ACB generation.

**# DMBS REFERENCE**
If the member is a PSB, this value is the number of DMBs that each PSB refers to.

**# PSBS REFERENCING**
If the member is a DMB, this value is the number of PSBs that contain references to it.

**COMMENTS**
The following comments might be shown:
- ALIAS
- PSB REFERS TO MISSING DMB
- NO DBDS REFERENCED
- PSB REFERS TO INVALID DMB
- DMB NOT REFERENCED BY PSB

When GENDATE=YES is specified in the LISTLIB command, the DETAIL section contains the following additional fields.

**ACBGEN DATE**
The column shows the date when the ACB member was generated.

**ACBGEN TIME**
The column shows the time when the ACB member was generated.
The SUMMARY section displays the count of DMBs and PSBs found in the specified library, and the number of references to DBDs by all PSBs.

**Distribution of Member Sizes report**

This report provides a frequency distribution of DMB and PSB sizes for all members found in the specified input library.

This report is generated in the SYSPRINT data set. However, when GENDATE=YES is specified on the LISTLIB command, this report is generated in the FABQRPT data set instead of the SYSPRINT data set.

This report is generated only for ACB libraries. If the input for the utility is a DBD library or a PSB library (LIBTYPE=DBD or LIBTYPE=PSB), this report is not generated.

The RANGE column specifies bytes.

An example of this report is shown in the following figure.

---

<table>
<thead>
<tr>
<th>RANGE</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 511</td>
<td>9</td>
</tr>
<tr>
<td>512 - 1,023</td>
<td>8</td>
</tr>
<tr>
<td>1,024 - 1,535</td>
<td>3</td>
</tr>
<tr>
<td>2,048 - 2,559</td>
<td>1</td>
</tr>
<tr>
<td><strong>TOTAL BYTES</strong></td>
<td><strong>14,696</strong></td>
</tr>
</tbody>
</table>

**DMBS**

---

<table>
<thead>
<tr>
<th>RANGE</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 1,023</td>
<td>2</td>
</tr>
<tr>
<td>4,096 - 5,119</td>
<td>1</td>
</tr>
<tr>
<td>10,240 - 11,263</td>
<td>1</td>
</tr>
<tr>
<td><strong>TOTAL BYTES</strong></td>
<td><strong>16,688</strong></td>
</tr>
</tbody>
</table>

**PSBS**

---

_Figure 163. Distribution of Member Sizes report_

**Distribution of PSB Workarea Sizes report**

This report helps you in determining the size of the buffer pool that holds PSBs in the online environment. The distribution interval is in 1 K increments. The total bytes at the bottom of the report show the amount of memory required to hold all PSBs.
This report is generated in the SYSPRINT data set. However, when GENDATE=YES is specified on the LISTLIB command, this report is generated in the FABQRPT data set instead of the SYSPRINT data set.

This report is generated only for ACB libraries. If the input for the utility is a DBD library or a PSB library (LIBTYPE=DBD or LIBTYPE=PSB), this report is not generated.

The following figure shows an example of the Distribution of PSB Workarea Sizes report.

---

<table>
<thead>
<tr>
<th>RANGE</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 1,023</td>
<td>2</td>
</tr>
<tr>
<td>1,024 - 2,047</td>
<td>2</td>
</tr>
</tbody>
</table>

TOTAL BYTES = 4,080

---

Figure 164. Distribution of PSB Workarea Sizes report

**Chronological History of ACBGENs report**

This report provides a chronological history of the ACBGEN activity for all the members in the specified library. A summary for each unique time stamp found is presented.

This report is generated in the SYSPRINT data set. However, when GENDATE=YES is specified on the LISTLIB command, this report is generated in the FABQRPT data set instead of the SYSPRINT data set.

This report is generated only for ACB libraries. If the input for the utility is a DBD library or a PSB library (LIBTYPE=DBD or LIBTYPE=PSB), this report is not generated.

Subsections:

- “Sample report”
- “Report field descriptions” on page 394

**Sample report**

The following figure shows an example of the Chronological History of ACBGENs report.
Report field descriptions

**DATE and TIME**
These columns show the time stamp used during the ACBGEN. It is applied to all DMBs and PSBs that were added to or replaced in the ACB library at that time.

**#PSB and #DMB**
These columns show the number of PSBs and DMBs added to or replaced in the ACB library during the ACBGEN at the specified date and time.

**Warning Messages report**
This report provides all the warning messages that the ACBLIB Analyzer utility program issues.

This report is generated in the SYSPRINT data set. However, when GENDATE=YES is specified on the LISTLIB command, this report is generated in the FABQRPT data set instead of the SYSPRINT data set.

---

### CHRONOLOGICAL HISTORY OF ACBGENS

```
<table>
<thead>
<tr>
<th>DATE</th>
<th>DATE</th>
<th>TIME</th>
<th>#PSB'S</th>
<th>#DMB'S</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015.021</td>
<td>01/21/2015</td>
<td>17:29:55</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.025</td>
<td>01/25/2015</td>
<td>09:03:22</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2015.032</td>
<td>02/01/2015</td>
<td>11:45:42</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.079</td>
<td>03/20/2015</td>
<td>16:32:57</td>
<td>1,177</td>
<td>6</td>
</tr>
<tr>
<td>2015.079</td>
<td>03/20/2015</td>
<td>19:29:34</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2015.082</td>
<td>03/23/2015</td>
<td>13:39:33</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.089</td>
<td>03/30/2015</td>
<td>10:27:19</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2015.090</td>
<td>03/31/2015</td>
<td>17:36:48</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2015.101</td>
<td>04/11/2015</td>
<td>10:23:03</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.101</td>
<td>04/11/2015</td>
<td>10:34:28</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2015.102</td>
<td>04/12/2015</td>
<td>13:27:51</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2015.103</td>
<td>04/13/2015</td>
<td>09:43:54</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.105</td>
<td>04/15/2015</td>
<td>07:07:41</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2015.108</td>
<td>04/18/2015</td>
<td>20:32:42</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.108</td>
<td>04/18/2015</td>
<td>20:38:53</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.108</td>
<td>04/18/2015</td>
<td>20:43:21</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.111</td>
<td>04/21/2015</td>
<td>15:18:03</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.111</td>
<td>04/21/2015</td>
<td>15:29:47</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.124</td>
<td>05/04/2015</td>
<td>13:19:57</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2015.124</td>
<td>05/04/2015</td>
<td>13:51:18</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Note: portions of this report omitted

```
2015.188 07/07/2015 13:59:04 1 0
2015.266 09/23/2015 15:26:58 0 6
2015.267 09/24/2015 11:37:01 6 0
```

**NUMBER OF UNIQUE ACBGEN TIMESTAMPs = 140**

---

**Figure 165. Chronological History of ACBGENs report**
This report is generated only for ACB libraries. If the input for the utility is a DBD library or a PSB library (LIBTYPE=DBD or LIBTYPE=PSB), this report is not generated.

The following figure shows an example of the Warning Messages report.

+-------------------------------+  
| WARNING MESSAGES              |  
|-------------------------------+

FABQ1031W DMB 'BEALORD' REFERENCED BY PSB 'PE4YOINQ' IS NOT IN ACBLIB
FABQ1031W DMB 'BEALPART' REFERENCED BY PSB 'PEACPPUR' IS NOT IN ACBLIB
FABQ1031W DMB 'C330INVC' REFERENCED BY PSB 'C330READ' IS NOT IN ACBLIB
FABQ1031W DMB 'DLC0R01' ' REFERENCED BY PSB 'VS1CDRMS' IS NOT IN ACBLIB

Figure 166. Warning Messages report
Chapter 12. MFS Reversal utility

The MFS Reversal utility converts Message Format Services (MFS) control blocks back into IMS MFS utility control statements.

Topics:
- “MFS Reversal utility overview” on page 398
- “Restrictions and considerations for the MFS Reversal utility” on page 399
- “Converting MFS control blocks to control statements” on page 400
- “JCL requirements for the MFS Reversal utility” on page 401
- “Control statements for the MFS Reversal utility” on page 403
- “Output from the MFS Reversal utility” on page 407
- “Important notes about the generated source” on page 416
MFS Reversal utility overview

The MFS Reversal utility converts MFS MID, MOD, DIF, and DOF control blocks back into IMS MFS utility control statements. The MFS Reversal utility is helpful in cases where the MFS source files are lost or compromised.

Note: MID refers to Message Input Descriptor, MOD to Message Output Descriptor, DIF to Device Input Format, and DOF to Device Output Format.

Subsections:
- “Function overview”
- “Program structure”

Function overview

The MFS Reversal utility builds a list of the selected MIDs and MODs and generates a list of all DIFs and DOFs that are associated with them. It then builds the source for the selected MIDs and MODs and linked DIF and DOF MFS control blocks and saves them as members of a partitioned data set.

If the input to MFS Reversal contains a single name ALL, then the utility builds the source for all MID, MOD, DIF, and DOF MFS control blocks of the IMS MFS format library.

It also generates a summary report of the relationships between MIDs/MODs and DIFs/DOFs, and relationships between MIDs and MODs of the specified MFS format library.

The MFS Reversal utility also provides a function to copy the selected MFS control blocks (MIDs, MODs, DIFs, DOFs) and their associated control blocks from the IMS format library to a user-specified partitioned data set. The utility also generates a report that summarizes the results of the copy process. You can use this function to back up the MFS control blocks.

Program structure

The MFS Reversal program consists of two load modules. One of them, called FABVRVRS, is the actual utility program. The second is a Device Characteristics Table called DFSUDT0x. The suffix x has a default value of A, but you can select another suffix using a control statement of the MFS Reversal utility. This module is fetched from your IMS RESLIB. A default Device Characteristics Table called FABVDVCT is provided with MFS Reversal in case you do not have the DFSUDT0x module. The DFSUDT0x table describes the default characteristics of 3270-An devices.
Restrictions and considerations for the MFS Reversal utility

Certain restrictions and considerations apply when using the MFS Reversal utility.

Restrictions

The copy function of the MFS Reversal utility does not support alias members. When alias members exist in the FORMAT DD, the copy function skips these members and continues processing. When alias members exist in the COPYFMT DD, the copy function ends with an error message. Before you run the copy function, ensure that the partitioned data set that is specified by the COPYFMT DD does not contain alias members.

Considerations

The MFS Reversal utility can execute while IMS Online is active. However, during the execution of the utility, the status of the format library must not be changed through the use of the IMS Online MODIFY command. Also, the JCL of the IMS Control Region and that of the utility must allow sharing the IMS MFS format library.

The MFS Reversal utility makes certain assumptions when generating source. You might need to modify the generated source before you use the source as input to the MFS Language utility. See “Important notes about the generated source” on page 416 for information about the generated source.
Converting MFS control blocks to control statements

To convert MFS control blocks (MIDs, MODs, DIFs, and DOFs) back into the IMS MFS utility control statements or to copy MFS control blocks by using the MFS Reversal utility, you must prepare the JCL for MFS Reversal and submit the job.

About this task

Sample JCL for the MFS Reversal utility is in the SHPSJCL0 library, member FABVIVP. You can modify this sample JCL and then use it to run the utility.

Procedure

1. In the MFS Reversal JCL, code the EXEC statement and DD statements.
   See “JCL requirements for the MFS Reversal utility” on page 401.
2. In the SYSIN data set, code the control statements for the MFS Reversal utility.
   See “Control statements for the MFS Reversal utility” on page 403.
3. Submit the job.
4. Check the output data sets that are generated.
   See “Output from the MFS Reversal utility” on page 407.

What to do next

The MFS Reversal utility makes certain assumptions when generating source. You might need to modify the generated source before you use the source as input to the MFS Language utility. See “Important notes about the generated source” on page 416 for information about the generated source.
JCL requirements for the MFS Reversal utility

When you code the JCL to run the MFS Reversal utility, include the EXEC statement and appropriate DD statements.

Subsections:
- “JCL example”
- “EXEC statement”
- “DD statements”

JCL example

An example of the JCL that is required for MFS Reversal is shown in the following figure.

```
//stepname EXEC PGM=FABVRVRS
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//SYSPRINT DD SYSPRINT**,DCB=BLKSIZE=133
//SYSOUT DD SYSPRINT**,DCB=BLKSIZE=133
//FORMAT DD DSN=ims.format,DISP=SHR
//MFSSRCE DD DSN=user.mfs.source,DISP=SHR
//SYSIN DD *
```

Figure 167. Example of MFS Reversal JCL (FABVRVRS JCL)

Modify the fields shown in lower case to reflect your operating environment.

EXEC statement

This statement invokes the MFS Reversal utility, FABVRVRS. The statement must be in the format shown:

```
//stepname EXEC PGM=FABVRVRS
```

DD statements

Code the following DD statements to identify the source of input and the placement of output information:

STEPLIB DD

This library contains the FABVRVRS utility program in its executable form. Change the name HPS.SHPSLMD0 to reflect the name used at your site. Concatenate the IMS RESLIB as part of the STEPLIB. Change the name IMSVS.SDFSRESL to the name used at your site.

SYSPRINT DD

This sequential data set contains the cross-reference report from the FABVRVRS program. The data set can be the JES spool data set or a standard sequential data set.

SYSOUT DD

This is the sequential data set that contains activity logs and any errors encountered during the execution of FABVRVRS. The data set can be the JES spool data set or a standard sequential data set.
FORMAT DD
This library contains the IMS MFS formats for which the source is to be generated. Check the DISP= parameter if this library is also used in the IMS Online Control Region JCL.

If two or more data sets are concatenated to the FORMAT DD, MFS Reversal processes only the first data set.

MFSSRCE DD
This data set is a partitioned data set and contains the IMS MFS control block source statements generated by the utility. The LRECL of this data set must be 80. The block size can be any valid multiple of 80.

SYSIN DD
This is the control data set for this program.

The record format is fixed blocked, and the logical record length is 80. The block size, if coded, must be a multiple of 80. SYSIN contains the control statements for the MFS Reversal utility.

Related reading: For the format of the control statements, see “Control statements for the MFS Reversal utility” on page 403.

COPYFMT DD
This DD statement is required when the copy function is used. This partitioned data set contains the MFS control blocks that are copied by the copy function. Attributes of the data set must be same as the MFS format library. For information about the MFS format library, see the IMS Installation.

COPYPRT DD
This optional DD statement is used when the copy function is used. This data set, if provided, contains the copy report that summarizes the results of the copy process. The record format is fixed-blocked, and the logical record length is 133. The block size, if coded, must be a multiple of 133.
Control statements for the MFS Reversal utility

Control statements for the MFS Reversal utility specify the functions to be performed; they must be placed in the SYSIN data set.

Subsections:
- “Syntax rules”
- “Control statement keywords”

Syntax rules

The control statements for the MFS Reversal utility consist of keywords and operands arranged in a simple fixed format.

- Control statements can start anywhere after the second column.
- Statements with an asterisk (*) in column 1 are treated as comments.
- The comment statement is allowable between continuous statements.

Control statement keywords

The following control statement keywords can be used for the MFS Reversal utility:

DVCTBL

The optional DVCTBL statement specifies the suffix x to build the device table called DFSUDT0x. This table contains the device characteristics of symbolically referenced devices 3270-An. The default suffix is the character A. You can specify only one DVCTBL statement for each execution of the MFS Reversal program.

The operand for the DVCTBL statement is a single character.

The following example illustrates the specification of the DVCTBL statement.

DVCTBL L

OPTION

The optional OPTION statement specifies whether to create or suppress the optional MFS Reversal utility outputs. The following outputs are controlled by the OPTION statements:

- MFS utility program output statements
- Cross-reference report
- Copies of MFS control blocks and their associated control blocks

The OPTION statement supports three parameters. You can specify only one parameter for each OPTION statement. To specify multiple parameters, code multiple OPTION statements.

NORVRS

This option suppresses the creation of the MFS utility program output statements. However, a report of all formats used by the MIDs and MODs that are specified on the SELECT statement is produced.

NOXRPT

This option suppresses the creation and printing of the cross-reference report.
COPY=YES | NO
This option specifies whether the MFS Reversal utility copies the selected MFS control blocks and their associated control blocks from the IMS format library to the user-specified partitioned data set.

YES Specifies the use of the copy function. The MFS Reversal utility copies the control blocks to the user-specified partitioned data set that is specified in the COPYFMT DD. When YES is specified, you can specify the REPLACE option.

REPLACE=YES | NO
This option specifies whether the copy function replaces the existing MFS control blocks in the user-specified partitioned data set that is specified in the COPYFMT DD. Specify this option after the COPY=YES option, on the same line, separated by a comma (.). Blank characters are not permitted between the options.

YES Specifies that the copy function replaces the existing MFS control blocks.

Tip: When the partitioned data set contains many members to be replaced, specifying REPLACE=YES might degrade the performance and might also cause a shortage of data set space. If performance degradation or space shortage is a concern, remove the existing members and redefine the data set before you rerun the job.

NO Specifies that the copy function does not replace the existing MFS control blocks. When a member with the same name is found, the copy function skips copying that member and processes the next member. REPLACE=NO is the default.

NO Specifies that the copy function is not used. COPY=NO is the default.

The following examples illustrate the specification of the OPTION statement.

Example 1: To suppress the creation of the MFS utility program output statements:

OPTION NORVRS

Example 2: To suppress the creation and printing of the cross-reference report:

OPTION NOXRPT

Example 3: To enable the copy function and replace the existing MFS control blocks:

OPTION COPY=YES,REPLACE=YES
Example 4:
To specify all the options:

```
OPTION NORVRS
OPTION NOXRPT
OPTION COPY=YES,REPLACE=YES
```

**SELECT**
The SELECT statement is a required statement that specifies that the MFS Reversal program re-create the control statements for the MFS utility from the control blocks named as operands on the SELECT statement.

When the copy function is enabled by the OPTION statement, the MFS control blocks that are named as operands on the SELECT statement and their associated MFS control blocks are copied to a partitioned data set that is specified in the COPYFMT DD. The copy function copies the following control blocks:

- MID and MODs that are referenced by the MID and MODs that are specified on the SELECT statement
- DIFs and DOFs that are associated with the MID and MODs that are specified on the SELECT statement
- MID and MODs that reference the DIFs and DOFs that are to be copied

You can specify only one SELECT statement for each execution of the MFS Reversal program.

Either of the following operands can be specified:

- `member_name`
  Specify the member names of the MID and MOD from which the source is to be generated. Only the names of the MID and MOD can be specified. The names of the DIF and DOF are not included in the list of names, as the MFS Reversal program gets these names from the corresponding MID and MOD control blocks.
  
  At least one member must be selected. To specify multiple member names, the names must be separated by commas and must end with a blank. The list of names can be continued on the next line by placing a comma after the last name on the current line and continuing with names on the next line. If a comma is the last character on a line, it is assumed that the selection continues on the next line. A name must be contained on a single line.

- `ALL`
  This option specifies all MID and MODs.

The following examples illustrate the specifications of the SELECT statement.

**Example 1:**
```
SELECT MID1,MID2,MOD1,MOD2
```

**Example 2:**
```
SELECT MID1,
    MID2,
    MOD1
```

**Example 3:**
```
SELECT ALL
```

**EXCLUDE**
The EXCLUDE statement works oppositely compared to the SELECT statement. The EXCLUDE statement specifies that the MFS Reversal
program does not re-create the control statements for the MFS utility from the control blocks that are named as operands on the EXCLUDE statement. The MFS Reversal utility skips processing for each member of a MID, MOD, DIF, and DOF that is specified on the EXCLUDE statements.

If a name is specified on the SELECT statement as well as on the EXCLUDE statement, the EXCLUDE statement has precedence.

When MIDs or MODs are specified on the EXCLUDE statement and OPTION NOXRPT is not specified, the NXT= columns for the MIDs or MODs are blank and the MIDs or MODs are not displayed in the REFERENCED BY columns in the MID/MOD XREF report. Similarly, these columns are blank for the MIDs and MODs that refer to the DIFs and DOFs that are specified by the EXCLUDE statement.

Any specification made to the EXCLUDE statement does not affect the behavior of the copy function.

Up to 511 EXCLUDE statements can be specified for each execution of the MFS Reversal program.

The operand for the EXCLUDE statement is a name of an MID, MOD, DIF, or DOF from which the source is not to be generated. The name must be 1 - 8 characters and must be specified after the EXCLUDE keyword with one blank between. Only one name is allowed per EXCLUDE statement. DIF and DOF names start with two non-alphabetical characters and can include lower case characters. To refer to a DIF/DOF, use the third to eighth character of the name, in uppercase only, and prefix the name with **.

The following example illustrates the specification of the EXCLUDE statement.

```
SELECT ALL
EXCLUDE ABCD MID or MOD name
EXCLUDE **FBCUL DIF or DOF name
```
Output from the MFS Reversal utility

Output from the MFS Reversal utility consists of the SYSOUT data set, the SYSPRINT data set, the MFSSRCE data sets, the COPYFMT data set, and the COPYPRRT data set.

SYSOUT data set

The SYSOUT data set contains the messages issued by MFS Reversal.

The following figure shows messages that are generated in the SYSOUT data set.

SYSPRINT data set

The SYSPRINT data set contains an MFS Reversal report that summarizes the cross-reference information between the message descriptors and the device format, and the cross-reference information between message descriptors.

This report includes the names of the MIDs and MODs that are specified on the SELECT statement, and the names of the DIFs and DOFs that are referenced by each MID and MOD. If ALL is specified on the SELECT statement, then all MID and MOD names in the format library and their referenced DIFs and DOFs are reported.

When the analysis of a format library member fails, information about the member is displayed before the MID/MOD XREF report.

Subsections:

• “Sample report”
• “Report field descriptions” on page 408

Sample report

The following report shows an example of the MID/MOD XREF report.
## Report field descriptions

The following explanations refer to some of the fields in the MID/MOD XREF report.

### MSGNAME

This column shows the MID or MOD name.

(I) Input; indicates a MID

(O) Output; indicates a MOD

### SOR=

This column gives the format name referred to by the MID or MOD.

### DCODE

This column shows the device code and is the first character in the DIF/DOF name.

### FCODE

This column shows the feature code and is the second character in the DIF/DOF name.

### FEATURE

This column shows the interpretation of the FCODE.

### NXT=

This column contains the names of other MIDs or MODs that this MID or MOD refers to. If this MID or MOD refers to no MIDs or MODs, this column is blank.

### REFERENCED BY

This column contains the names of other MIDs or MODs that are specified on the SELECT statement and that refer to this MID or MOD. If no MIDs or MODs refer to this MID or MOD, this column is blank.

### WARNING:

A warning message is given when a message member (MID or MOD) is selected for source generation and the program finds other members in the

---

<table>
<thead>
<tr>
<th>MSGNAME</th>
<th>SOR</th>
<th>DCODE</th>
<th>DEVICE</th>
<th>FCODE</th>
<th>FEATURE</th>
<th>NXT</th>
<th>REFERENCED BY</th>
</tr>
</thead>
<tbody>
<tr>
<td>M03270B</td>
<td>(O)</td>
<td>F3270B</td>
<td>02</td>
<td>3270,2</td>
<td>7F</td>
<td>FEAT-IGNORE</td>
<td>M03270K M03270K</td>
</tr>
<tr>
<td>M13270K</td>
<td>(I)</td>
<td>F3270K</td>
<td>42</td>
<td>3270-A02</td>
<td>C7</td>
<td>FEAT=(CARD,PFK,PEN)</td>
<td>M03270K M13270J M13270K M1360B</td>
</tr>
<tr>
<td>M03270K</td>
<td>(O)</td>
<td>F3270K</td>
<td>42</td>
<td>3270-A02</td>
<td>C7</td>
<td>FEAT=(CARD,PFK,PEN)</td>
<td>M13270K</td>
</tr>
<tr>
<td>M03270J</td>
<td>(O)</td>
<td>F3270K</td>
<td>42</td>
<td>3270-A02</td>
<td>C7</td>
<td>FEAT=(CARD,PFK,PEN)</td>
<td>M13270K</td>
</tr>
<tr>
<td>M1360B</td>
<td>(I)</td>
<td>F1360B</td>
<td>08</td>
<td>FIN</td>
<td>7F</td>
<td>FEAT=IGNORE</td>
<td>M03270K</td>
</tr>
<tr>
<td>M1710B</td>
<td>(I)</td>
<td>D1710B</td>
<td>0C</td>
<td>3C51</td>
<td>01</td>
<td>FEAT=1</td>
<td>M03270K</td>
</tr>
<tr>
<td>M03270C</td>
<td>(O)</td>
<td>F3270C</td>
<td>02</td>
<td>3270,2</td>
<td>7F</td>
<td>FEAT=IGNORE</td>
<td>M13270C</td>
</tr>
<tr>
<td>M05C51C</td>
<td>(I)</td>
<td>F3270C</td>
<td>02</td>
<td>3270,2</td>
<td>7F</td>
<td>FEAT=IGNORE</td>
<td></td>
</tr>
</tbody>
</table>

---

**Figure 169. MID/MOD XREF report**
format library that refer to the same format name as the selected member. The cross-reference information between MID and MOD is not shown in the warning messages.

In the sample report, member MO3270C was selected but the program found members MI3270C and MOSCS1C that also reference the same FMT named F3270C. For completeness, consider running the utility again to include these message names. In this subsequent run of the utility, only those message members need be selected that were incomplete along with the names in the warning list.

Notes:
1. When MIDs or MODs are specified on the EXCLUDE statement, the NXT= column is blank.
2. When MIDs or MODs are specified on the EXCLUDE statement, MIDs or MODs that are referenced by the specified MIDs or the MODs are not displayed in the REFERENCED BY column.
3. When DIFs and DOFs are specified on the EXCLUDE statement, the NXT= column and the REFERENCED BY column are blank for the MIDs and MODs that refer to these DIFs and DOFs.
MFSSRCE data set

This partitioned data set contains the IMS MFS control block source statements generated by the utility.

You can optionally modify these source statements and then use them as input to the IMS MFS Language utility. This utility creates MFS control blocks (MIDs, MODs, DIFs, and DOFs).

Subsections:
- “How the control blocks are converted to source code”
- “Naming conventions”
- “Example of created source” on page 41

How the control blocks are converted to source code

A brief discussion of how MFS Reversal converts the format names found in the control blocks back to source code follows:

A format is referenced by a MID or MOD by the 3rd through 8th character of the format name. Depending on the device characteristics and features of the device from where the message is received, IMS Message Formatting Services retrieves the correct format control block. The first character of the control block name refers to the device and the second to the feature. One or more formats may exist having the same 3rd through 8th characters of the format name. Source statements are created for each device/feature combination occurring in a format (FMT) as well as the source for the MID, MOD, or both that reference this format.

The following example illustrates this:

If TESTMOD is a selected modname for which the source is to be created, and TESTMOD references xyTSTFMT, where x is a device code and y is a feature code, then a member is created in the MFSSRCE output library with the name TSTFMT. This member will have the FORMAT definitions (various DEV, DIV, DPAGE, DFLD, and similar statements) for all possible xy occurrences with the TSTFMT name and the MOD definitions for TESTMOD.

Note: When building the source, a check is made of the time stamp that appears in the MID, MOD, DIF, and DOF control blocks. The time stamp in the MID and associated DIF must match. So must the time stamps in the MOD and DOF. If the time stamp check fails, the selected MID or MOD is not processed. An informational message is written to the SYSPRINT data set.

Naming conventions

The FORMATs and the MID/MODs that are created by the MFS Reversal utility generate their own names and labels whenever possible.

The naming conventions followed are:

DPAGE label
Each DPAGE label is in the form DPAxxxx, where xxxx is a number in the range of 1 - 99999. The number is incremented by 1 for each new DPAGE in the DEV definition.
DFLD label
Each DFLD label is in the form DLnnnnnnn, where nnnnn is the decimal equivalent of an internal offset.

CURSOR name
Each cursor name is of the form CSnnnnnn, where nnnnn is the decimal equivalent of an internal offset.

MFLD label
Each MFLD label used in a conditional predicate on the LPAGE statement has the form CONDnnnnnnn, where nnnnn is a number in the range of 1 - 9999. The number is incremented each time an MFLD label is created within MSG.

MFLD name
An MFLD name is generally the label of a DFLD statement. However, if a field is not known to the MID, it is given the name NNxxxxxx, where xxxxx is a number in the range of 1 - 999999. The number is incremented by 1 for each new occurrence of an unknown field to the MID. If a field is not known to the MOD, it is given the name NN00OSSC.

PFK name
An MFLD name for PFKey input has the name PFK00001.

CARD name
An MFLD name for Magnetic Card Reader input has the name CRD00001.

PEN name
MFLD name for PEN input has the format PENxxxxx, where xxxxx is a number in the range of 1 - 99999. The number is incremented by 1 for each new PEN name.

ACTVPID name
MFLD name for active PID. The name has the format ACTVPIDDdd, where dd is a number in the range of 1 - 99. The number is incremented each time such a name is created.

OPCTL TABLE label
Each OPCTL TABLE is given a name having the format OPCTLzzz, where zzz is a number in the range of 1 - 999. The number is incremented by 1 for each new TABLE statement.

IF label
Each IF statement has a label of the form IFTtttttt, where tttttt is a number in the range of 1 - 999999. The number is incremented by 1 for each new IF statement creation.

Example of created source
The following figure illustrates source created for MO3270A and MI3270A. This output could have been created by submitting the following control cards to the MFS Reversal utility:

DVCTBL Q
SELECT MO3270A,MI3270A

In this example, the MFS Reversal utility encountered a member X'027FF3270A in the format library. It then deconstructed this to create a format name of F3270A (which is the label on the FMT statement), a TYPE=(3270,2) and a FEAT=IGNORE.
The MFS Reversal utility combines all the members that have the same FORMAT name (suffix portion of xyF3270A) and creates a single member in MFSSRCE under the FORMAT name F3270A.

Related reading:
- See “Device and feature code tables” on page 530 to understand how the hex values X'02' and X'7F' in the format names are interpreted.
- See “Important notes about the generated source” on page 416 for additional information about the generated source.

The following figure shows an example of the source that is created by MFS Reversal.
COPYFMT data set

When the copy function is enabled by the OPTION statement, this partitioned data set contains the MFS control blocks that are specified on the SELECT statement and their associated MFS control blocks. These control blocks are copied from the IMS format library that is specified in the FORMAT DD.
When this partitioned data set contains alias members, which are not supported by the copy function, the program issues message FABV0065E and ends the copy process. When the copy process is ended for this reason, no MFS control blocks are copied to this partitioned data set.

**COPYPRT data set**

When the copy function is enabled by the OPTION statement, this data set contains a copy report, which summarizes the copy process.

**Subsections:**
- “Sample report: OPTION COPY=YES,REPLACE=NO”
- “Sample report: OPTION COPY=YES,REPLACE=YES” on page 415

**Sample report: OPTION COPY=YES,REPLACE=NO**

The following figure shows an example of the Copy report when OPTION COPY=YES,REPLACE=NO is specified.

---

The following explanations refer to the fields in the Copy report that is issued when OPTION COPY=YES,REPLACE=NO is specified.

**NUMBER OF COPIED MEMBERS**

The number of members that were copied.

**NUMBER OF COPY SKIPPED MEMBERS**

The number of members that were skipped.

**THE FOLLOWING MEMBERS WERE COPIED**

A list of the members that were copied.

**THE FOLLOWING MEMBERS WERE SKIPPED**

A list of the members that were skipped. Skipped members are shown when OPTION COPY=YES,REPLACE=NO is specified in the SYSIN DD and the members to be copied exist in the partitioned data set that is specified by the COPYFMT DD.

The following figure shows an example of the Copy report when one or more members could not be copied. When members that could not be copied are displayed in the report, identify the cause of errors from the messages that are recorded in the SYSOUT data set.
The following explanations refer to some of the fields in the Copy report when copy failed for some members.

**NUMBER OF COPY FAILED MEMBERS**
The number of members that could not be copied.

**THE FOLLOWING MEMBERS COULD NOT BE COPIED**
A list of the members that could not be copied.

**Sample report: OPTION COPY=YES,REPLACE=YES**

The following figure shows an example of the Copy report when OPTION COPY=YES,REPLACE=YES is specified.

The following explanations refer to some of the fields in the Copy report that is issued when OPTION COPY=YES,REPLACE=YES is specified.

**NUMBER OF REPLACED MEMBERS**
The number of members that were replaced.

**THE FOLLOWING MEMBERS WERE REPLACED**
A list of the members that were replaced. Replaced members are shown when OPTION COPY=YES,REPLACE=YES is specified in the SYSIN DD and the members to be copied exist in the partitioned data set that is specified in the COPYFMT DD.
Important notes about the generated source

The MFS Reversal utility makes certain assumptions when generating source.

The MFS Reversal utility makes the following assumptions:

- If OPT=3, the LTH value is taken from the DIF/DOF; otherwise it is taken from the MID/MOD.
- If the MSG statement of a MID or MOD has the NXT keyword, it is propagated to each LPAGE statement in the MID/MOD.
- If OPT=3 and the MFLD is not known to the DOF, then the LTH value on the MFLD statement is also unknown.
- If any of the formats uses lower case characters, then an ALPHA statement must be included in the output of the Reversal utility for the specific format - if this output is to be processed by the MFS Language utility.

**Attention:** The source generated by the MFS Reversal utility might need to be modified for subsequent processing by the IMS MFS Language utility.

- The DSCA value is taken from the DOF. This might not be the same as what the original source data for the DOF contained, because the MFS Language utility checks the value specified for the DSCA and, if necessary, modifies it.
- The WIDTH value for output devices is always the default value. You might need to change this value if the output of MFS Reversal is to be processed by the MFS Language utility.

**Attention:** The source generated by the MFS Reversal utility might need to be modified for subsequent processing by the IMS MFS Language utility.

- OUTL'0E' is reported as RIGHT,OVER,LEFT.
- EGCS'xx' specification is reported in the form PX'xx'.
- If the EATTR has a value of PX'A', it is reported in hex form as PX'C1'.
- The LTH keyword is not shown on a literal DFLD.
- The page size value on the PAGE keyword of the DEV statement is always the default page size. You might need to change this value if the output of the Reversal utility is to be processed by the MFS Language utility.

**Attention:** The source generated by the MFS Reversal utility might need to be modified for subsequent processing by the IMS MFS Language utility.

- Finance terminals 3600 and 36DS are reported as Device Type FIDS.
- Finance terminals 36DS3, 36DS4, 36DS7 are reported as Device Types FIDS3, FIDS4, and FIDS7, respectively.
- Finance terminals 36JP, 36FP, and 36BP are reported as FIJP, FIFP, and FIBP, respectively.
- For DPMA output, HDRCTL always has a length of 7.
- For DPMA devices, an RCD statement is implied after the first PPAGE statement. However subsequent PPAGE statements have RCD statements following each PPAGE statement.
- If RCDCTL was defined as RCDCTL=(,SPAN), but there were no fields that spanned a line, then the result is RCDCTL=(xxx,NOSPAN), where xxx is the maximum record length.
- If OPTIONS=SIM was defined in the format and there are no fields with simulated attributes, then the result is OPTIONS=NOSIM2.
- For DPMB devices, if OPTIONS=DPAGE is specified on the DIV statement and PPAGE statements are also present, then the PPAGE name in the FMH header is given to the DPAGE statement and all PPAGE statements are ignored.
• For SCS1 devices and TYPE=INPUT, the first DFLD of the DIF starts at position (1,1).
• If ENDMG or (BGNMSG,ENDMSG) or (BGNPP,BGNMSG,ENDMSG) is the EJECT option, the ENDMG is reported as ENDPP.
• The DEFN and SPACE options of the PAGE keyword for SCS1 devices cannot be resolved.
• SLDI is always reported in the SLDP form.
• If OPTIONS=MSG, any PPAGE definition is ignored.
• If a literal definition that includes Double Byte Character Set (DBCS) characters is continued over multiple lines, the first byte of the DBCS character could be in column 71 and the second byte of the character could be at the beginning of the next line. In this case, even the DBCS characters are not enclosed with SO and SI control characters, this line continuation format conforms to the continuation rules of MFS source, and therefore, the source can be processed by the IMS MFS Language utility without modification.
Chapter 13. MFS Compare utility

The MFS Compare utility compares IMS MFS control blocks that reside in two different MFS format libraries.

Topics:
- “MFS Compare utility overview” on page 420
- “Considerations for the MFS Compare utility” on page 422
- “Keywords used in comparisons” on page 423
- “Comparing MFS control blocks” on page 428
- “JCL requirements for the MFS Compare utility” on page 429
- “Control statements for the MFS Compare utility” on page 431
- “Output from the MFS Compare utility” on page 433
MFS Compare utility overview

The MFS Compare utility compares IMS MFS control blocks that reside in two different MFS format libraries. The utility first decodes two sets of IMS MFS control blocks into IMS MFS source statements, and then compares the two sets of source statements.

This function is useful when you want to validate that what you are running corresponds to what is in your source library. You would do this by first running the IMS MFS Language utility using your source statements as input to create a set of MFS control blocks that reside in a separate MFS format library. Then run the Compare utility to see if the two sets of control blocks match.

Subsections:
- “Function overview”
- “Program structure” on page 421

Function overview

The MFS Compare utility creates IMS MFS source statements from MFS control blocks that reside in two MFS format libraries. The utility uses the Message Input Descriptors (MIDs) and Message Output Descriptors (MODs) specified as input to build and store the source Formats (FMTs) as members of work partitioned data sets. If ALL is specified as input, then the utility builds the source for all MID, MOD, DIF, and DOF IMS MFS control blocks.

As part of this process, the utility first builds a list of the selected MIDs and MODs and generates a list of all DIFs and DOFs associated with the MIDs and MODs.

Then the source is generated and stored as members of two partitioned data sets. Each member represents an entire Format (FMT). The FMT consists of multiple Format Control Blocks (FCBs). An FCB can be an MSG, DEV, PDB, or TABLE control block.

Note: When generating the IMS MFS utility control statements, a check is made of the time stamp that appears in the MID, MOD, DIF, and DOF control blocks. The time stamp in the MID and associated DIF must match. So must the time stamps in the MOD and DOF. If the time stamp check fails, the selected MID or MOD is not processed. An informational message is written to the SYSPRINT data set.

When the source is generated and stored, the utility compares each format in the first partitioned data set with the corresponding format in the second partitioned data set and reports any differences. The comparison is performed FMT by FMT. Only FMTs having the same name in both the libraries are compared. FCBs within the format are compared if their names are the same. An informational message is issued when:
- an FMT exists in one library but not in the other
- an FCB exists in a format in one library but not in the other

Multiple DIV statements can exist in a format for the same DEV. If that is the case, the comparison is done at the DIV level.

The MFS Compare utility compares the FCB parameters even if they are unused for some IMS versions and releases.
Program structure

The MFS Compare utility consists of two load modules. One of them, called FABVCMPR, is the actual utility program. The second is a Device Characteristics Table called DFSUDT0x. The suffix $x$ has a default value of $A$, but you can select another suffix using a control statement of MFS Compare. This module is fetched from your IMS RESLIB. A default Device Characteristics Table called FABVDVCT is provided with MFS Compare in case you do not have the DFSUDT0x module.
Considerations for the MFS Compare utility

Certain considerations apply when you use the MFS Compare utility.

The MFS Compare utility can execute while IMS Online is active. However, during the execution of the utility, the status of the format library must not be changed through the use of the IMS Online MODIFY command. Also, the JCL of the IMS Control Region and that of the utility must allow sharing the IMS MFS format libraries.
Keywords used in comparisons

The MFS Compare utility compares the keywords that are in the generated source statements.

The following keywords are compared for each format control block (MSG, DEV, PDB, and TABLE):

**MSG**
- **TYPE** Indicates input (MID) or output (MOD)
- **SOR** Names the corresponding DIF or DOF
- **OPT** Represents the message option number
- **NXT** Links this MID to the next MOD or this MOD to the next MID
- **PAGE** Indicates whether operator logical paging is provided
- **FILL** Indicates the fill character for output devices

**LPAGE**
- **SOR** Links this logical page to the corresponding DPAGE in the device format (DIF or DOF)
- **COND** Describes a conditional text for editing of this logical page
- **NXT** Links this MID to the next MOD or this MOD to the next MID if the logical page is processed
- **PROMPT** Defines a literal to be placed in a field when formatting the last logical page

**PASSWORD**
- Defines a password segment of one or more Message fields

**SEG**
- **EXIT** Describes the segment edit exit routine interface
- **GRAPHIC** Specifies whether uppercase translation is to occur

**MFLD**
- **DFLDNAME** Specifies the device field name from which data is extracted or into which data is placed
- **'LITERAL'** A value inserted in an input message
- **LTH** Specifies the length of the field
- **JUST** Specifies that the data field is to be left-aligned or right-aligned
- **ATTR** Specifies whether the application program can modify the 3270 attributes and extended attributes
- **FILL** Specifies the message field padding character
- **EXIT** Describes the field edit exit routine interface
- **SCA** Defines the system control area
DEV

TYPE  Specifies the device type and model number of a device using this format description
FEAT  Specifies the feature for this device or program group
MODE  Specifies the manner in which field scanning is to occur
FTAB  Specifies the field tab character used to terminate an input field
LDEL  Specifies the two characters used to determine if a record is to be discarded
PGE   Specifies the characteristics of a physical page
DSCA  Specifies the default system control area
PEN   Defines an input field name to contain literal data when an immediate light pen detection of a field with a space or null designator character occurs
CARD  Defines the input field name to receive operator identification card data when that data is entered
SYSMSG Specifies the label of the field in the DFLD statements that define the device field in which IMS system messages are to be displayed
PFK   Defines an input field name to contain program function key literal data or control function data/action
SUB   Specifies the character used by MFS to replace any X'3F' characters in the input data stream
PDB   Specifies the name of the Partition Descriptor Block to describe the partition set
WIDTH Specifies the maximum line width for this device type
FORMS Specifies a literal included on the output message
HTAB  Specifies the position where horizontal tab stops are placed
VT    Specifies that MFS inserts tab control characters at the specified locations
VTAB  Specifies the positions of top and bottom page margins
SLDI  Specifies the line density for an output message in lines per inch
SLDP  Specifies the line density for an output message in points per inch
VERSID Specifies the version ID

DIV

TYPE  Specifies the format type (INPUT, OUTPUT, or INOUT)
OPTIONS Specifies the exit routine to map data
COMPR
Requests MFS to remove trailing blanks from short fields, fixed-length fields, or all fields presented by the application program.

RCDTL
Specifies the maximum length of a transmission record.

NULL
Specifies whether MFS is to ignore, search, or replace trailing nulls in fields.

HDRCTL
Specifies the characteristics of the output message header.

RDPN
Permits the suggested return destination process name to be supplied in the input message MFLD referenced.

DPN
Specifies the destination process name.

PRN
Specifies the primary resource name.

RPRN
Permits the suggested return primary resource name to be supplied in the input message MFLD referenced.

OFTAB
Directs MFS to insert output field tab separators characters in the output data stream.

DPAGE
COND
Specifies a conditional test to be performed on the first input record.

FILL
Specifies a fill character for output device fields.

OFTAB
Directs MFS to insert the output field tab separator character specified on this DPAGE statement for the output data stream of the DPAGE being described.

CURSOR
Specifies the position of the cursor on a physical page.

MULT
Specifies that multiple physical page input messages are allowed for this DPAGE.

PD
Specifies the name of the partition descriptor of the partition associated with the DPAGE statement.

ACTVID
Specifies the name of an output field in the message containing the partition identification number of the partition to be activated.

ORIGIN
Specifies page positioning on the Finance display for each physical page defined.

SELECT
Specifies carriage selection for a FIFP device with FEAT=DUAL specified in the previous DEV statement.

PPAGE
Defines the beginning of a presentation page.

RCD
Can be used to influence the placement of DFLDs in records.
DFLD

'LITERAL'
Specifies a literal character string to be presented to the device

G'LITERAL'
Specifies an EGCS literal character string to be presented to the device

POS
Defines the first data position of this field in terms of line, column, and physical page of the display format

LTH
Specifies the length of the field

ATTR
Defines the display attributes of this field

OPCTL
Specifies the name of a table that is to be checked for operator control requests when this device field is received

SLDI
Specifies the line density for an output message in lines per inch

SLDP
Specifies the line density for an output message in points per inch

PASSWORD
Identifies this field as the location of the IMS password field for input messages

PEN
Specifies a literal to be selected or an operator control function to be performed when this field is detected

EATTR
Defines the extended attributes of this field

PDB

LUSIZE
Describes the physical size of the Logical Unit display for which the PDB is defined

SYSTEM
Specifies the partition name for displaying system messages

PAGINGOP
Specifies the option number for the partition page presentation algorithm

LUDEFN
Indicates whether the LUSIZE parameter in the PDB statement and the VIEWLOC parameter in the PD statements are specified in rows and columns or in pels

PD

PID
Specifies a partition identifier number for the partition

VIEWPORT
Specifies the size of the viewport for the partition

VIEWLOC
Specifies the location of the viewport on the display screen in terms of the distance offset from the upper-left of the screen
PRESPACE
Indicates the size of the presentation space buffer in row and columns

WINDOWF
Indicates the initial offset in rows of the top edge of the view window from the top of the presentation space

CELLSIZE
Indicates the number of horizontal and vertical pels in a character cell

SCROLLI
Indicates the number of rows that are scrolled when the scrolling function is used

TABLE

IF

DATA Specifies that the conditional operation be performed against the data received from the device for the field

LENGTH Specifies that the conditional operation is testing the number of characters entered for the field
Comparing MFS control blocks

To compare IMS MFS control blocks that reside in two different MFS format libraries by using the MFS Compare utility, you must create a set of MFS control blocks from your source statements, prepare and submit the JCL for the MFS Compare utility, and check the differences in the MFS Compare report.

About this task

Sample JCL for the MFS Compare utility is in the SHPSJCL0 library, member FABVIVP. You can modify this sample JCL and then use it to run the utility.

Procedure

1. Use the IMS MFS Language utility (DFSUPAA0) to create a set of MFS control blocks from your source statements.
   For the instructions to use the MFS Language utility, see IMS System Utilities.
2. In the MFS Compare JCL, code the EXEC statement and DD statements.
   See “JCL requirements for the MFS Compare utility” on page 429.
3. In the SYSIN data set, code the control statements for the MFS Compare utility.
   See “Control statements for the MFS Compare utility” on page 431.
4. Submit the job.
5. Check the Compare report that is generated in the SYSPRINT data set.
   See “Output from the MFS Compare utility” on page 433.

What to do next

After checking the Compare report, you might become aware of some discrepancies in your format libraries.

It is up to you to decide which values are appropriate for your site. When you have determined the values to use, you can run the MFS Reversal utility, modify the MFSSRCE source file to reflect the new values, and then regenerate the control blocks by using the IMS MFS Language utility.
JCL requirements for the MFS Compare utility

When you code the JCL to run the MFS Compare utility, include the EXEC statement and appropriate DD statements.

Subsections:
- "JCL example"
- "EXEC statement"
- "DD statements"

JCL example

An example of the JCL that is required for MFS Compare is shown in the following figure.

```
//stepname EXEC PGM=FABVCMPR
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMSVS.SDFSRESL,DISP=SHR
//SYSPRINT DD SYSOUT=*,DCB=BLKSIZE=133
//SYSOUT DD SYSOUT=*,DCB=BLKSIZE=133
//FORMAT DD DSN=ims.format1,DISP=SHR
//FORMAT2 DD DSN=ims.format2,DISP=SHR
//MFSSRCE DD DSN=&SOURCE1,DISP=(,DELETE),
//SPACE=(CYL,(5,5,20))
//MFSSRCE2 DD DSN=&SOURCE2,DISP=(,DELETE),
// SPACE=(CYL,(5,5,20))
//SYSSIN DD *
```

Figure 174. Example of MFS Compare JCL (FABVCMPR JCL)

Modify the fields shown in lower case to reflect your operating environment.

EXEC statement

This statement invokes the MFS Compare utility, FABVCMPR. The statement must be in the following format:

```
//stepname EXEC PGM=FABVCMPR
```

DD statements

Code the following DD statements to identify the source of input and the placement of output information:

STEPLIB DD

This library contains the FABVCMPR utility program in its executable form. Change the name HPS.SHPSLMD0 to reflect the name used at your site. Concatenate the IMS RESLIB as part of the STEPLIB. Change the name IMSVS.SDFSRESL to the name used at your site.

SYSPRINT DD

This sequential data set lists the results of the compare procedure. The data set can be the JES spool data set or a standard sequential data set.

SYSSOUT DD

This sequential data set contains activity logs and any errors encountered during the execution of the FABVCMPR program. The data set can be the JES spool data set or a standard sequential data set.
**FORMAT DD**

This library contains the first set of IMS MFS format control blocks for comparison. Change the name `ims.format1` to the name used at your site. Check the DISP= parameter if this library is also used in the IMS Online Control Region JCL.

If two or more data sets are concatenated to the FORMAT DD, MFS Compare processes only the first data set.

**FORMAT2 DD**

This library contains the second set of IMS MFS format control blocks for comparison. Change the name `ims.format2` to the name used at your site. Check the DISP= parameter if this library is also used in the IMS Online Control Region JCL.

If two or more data sets are concatenated to the FORMAT2 DD, MFS Compare processes only the first data set.

**MFSSRCE DD**

This data set is a partitioned data set and contains the MFS control block source statements generated by the MFS Compare utility from the FORMAT data set. The LRECL of this data set must be 80. The block size can be any valid multiple of 80.

A format is referenced by a MID or MOD by the 3rd through 8th character of the format name. Depending on the device characteristics and features of the device from where the message is received, IMS Message Formatting Services retrieve the correct format control block. The first character of the control block name refers to the device and the second to the feature. One or more formats might exist having the same 3rd through 8th characters of the format name. Source statements are created for each device/feature combination occurring in a format (FMT) as well as the source for the MID, MOD, or both that reference this format.

The following example illustrates this:

If TESTMOD is a selected modname for which the source is to be created, and TESTMOD references `xy/TSTFMT`, where `x` is a device code and `y` is a feature code, then a member is created in the MFSSRCE output library with the name TSTFMT. This member will have the FORMAT definitions (various DEV, DIV, DPAGE, DFLD, and similar statements) for all possible `xy` occurrences with the TSTFMT name and the MOD definitions for TESTMOD.

**MFSSRCE2 DD**

This data set is a partitioned data set and contains the MFS control block source statements as generated by the utility using the FORMAT2 data set. The LRECL of this data set must be 80. The block size can be any valid multiple of 80.

**SYSIN DD**

This data set contains the control statements for this program.

The record format is fixed blocked, and the logical record length is 80. The block size, if coded, must be a multiple of 80. SYSIN contains the control statements for the MFS Compare utility.

**Related reading:** For the format of the control statements, see “Control statements for the MFS Compare utility” on page 431.
Control statements for the MFS Compare utility

Control statements for the MFS Compare utility specify the functions to be performed; they must be placed in the SYSIN data set.

Subsections:
- "Syntax rules"
- "Control statement keywords"

Syntax rules

The control statements for the MFS Compare utility consist of keywords and operands arranged in a simple fixed format.
- Control statements can start anywhere after the second column.
- Statements with an asterisk (*) in column 1 are treated as comments.
- The comment statement is allowable between continuous statements.

Control statement keywords

The following control statement keywords can be used for the MFS Compare utility:

**DVCTBL**

The optional DVCTBL statement specifies the suffix x to build the device table called DFSUDT0x. This table contains the device characteristics of symbolically referenced devices 3270-An. The default suffix is the character A. The table resides in the IMS RESLIB. You can specify only one DVCTBL statement for each execution of the MFS Compare utility.

The operand for the DVCTBL statement is a single character.

The following example illustrates the specification of the DVCTBL statement.

```
DVCTBL L
```

**SELECT**

The SELECT statement is a required statement that specifies that the MFS Compare program re-create and compare the sources of the members that are named as operands on the SELECT statement. You can specify only one SELECT statement for each execution of the MFS Compare utility.

Either of the following operands can be specified:

*member_name*

Specify the member names of the MID and MOD from which the source is to be generated. Only the names of the MID and MOD can be specified. The names of the DIF and DOF are not included in the list of names, as the MFS Compare program gets these names from the corresponding MID and MOD control blocks.

At least one member must be selected. To specify multiple member names, the names must be separated by commas and must end with a blank. The list of names can be continued on the next line by placing a comma after the last name on the current line and continuing with names on the next line. If a comma is the last character on a line, it is assumed that the selection continues on...
the next line. If no line follows the current line, the comma signals the end of the statement. A name must be contained on a single line.

**ALL**  This option specifies all MIDs and MODs.

The following examples illustrate the specifications of the SELECT statement.

**Example 1:**
```
SELECT MID1,MID2,MOD1,MOD2
```

**Example 2:**
```
SELECT MID1, MID2, MOD1
```

**Example 3:**
```
SELECT ALL
```
Output from the MFS Compare utility

Output from the MFS Compare utility consists of the SYSOUT data set and the SYSPRINT data set.

SYSOUT data set

The SYSOUT data set contains the messages issued by the MFS Compare utility.

In the SYSOUT data set, the MFS Compare utility writes its messages as follows:

- In the first page, the messages that are issued while processing the first set of IMS MFS format control blocks (specified by the FORMAT DD) are printed.
- In the second page, the messages that are issued while processing the second set of IMS MFS format control blocks (specified by the FORMAT2 DD) are printed.

The following figure shows messages that are generated in the SYSOUT data set.

![Figure 175. Messages in the SYSOUT data set](image)

SYSPRINT data set

The SYSPRINT data set contains the compare report that is generated by MFS Compare.

The SYSPRINT data set contains fixed-length records of 133 bytes and a block size of 133 or a multiple of 133.

Subsections:

- “Sample report”
- “Report field descriptions” on page 434

Sample report

The following figure shows an example of the MFS Compare report.
This report is for illustration purposes only. Because the source is generated by a program, invalid keywords do not occur in the actual reports.

**Report field descriptions**

The following explanations refer to the MFS Compare sample report.

**Line 1** FORMAT is the DD name as required by the JCL. VNDR400.FORMAT is the data set name of the first format library whose members are to be compared.

```
FORMAT: VNDR400.FORMAT
FORMAT2: VNDR400.FORMAT
```

**Line 2** This header line specifies a LABEL STATEMENT and KEYWORD sequence for each format library.

```
LABEL STATEMENT KEYWORD
```

**Line 3** The format named ADDFMT is in the first format library but not in the second.

**Line 4** The format named CDCPI1 was compared and the compare was completed. Because no other keywords and parameters of this format are listed, the compare was successful.

**Line 5** The format named FISC1A is being compared.
Line 6  The FDC DEV0C7F of the format FISC1A is being compared. The value 0C in the device name indicates that the device is an SCS1 type device and the value 7F indicates that the features of this device are to be ignored.

**Related reading:** For a complete description of device codes and feature codes, see “Device and feature code tables” on page 530.

Line 7  The device input format DIVIN is being compared. The characters IN in the device name indicate that it is an input device.

Line 8  The DPAGE DPA00002 of the above DIV (DIVIN) is being compared.

Lines 9 and 10  The DFLD DL000034 of the above DPAGE was found in the second library, but not in the first. This DFLD had the parameters POS=(1,13) and LTH=5.

Line 11  An MSG format control block named MISC1A was found in the FMT named FISC1A in the second library, but not in the first.

Line 12  This line indicates the completion of the compare for the FMT FISC1A.

Line 13  The FMT FISC1B is not in the first library, but was found in the second library.

Line 14  The FCB FOSC1A is being compared.

Line 15  The FCB DEV0C7F in the first library has a FORMS parameter that is different from the FORMS parameter of the same named FCB in the second library.

Line 16  The FCB DEV0C7F of the FCB named FOSC1A in the first library has a parameter keyword XTAB not found in the corresponding DEV in the second library.

Lines 17 and 18  The FCB DEV0C7F of the FCB named FOSC1A in the second library has a parameter keyword HTAB not found in the corresponding DEV in the first library.

Line 19  The FCB MOSC1A in the current FMT is being compared.

Line 20  The LPAGE of MOSC1A is being compared.

Line 21  The SEG of the current LPAGE is being compared.

Line 22  The MFLD references an unknown DFLD named NN001079 in the first library. The same MFLD from the second library references the unknown field NN001001.

Line 23  Similar to line 22.

Lines 24, 25 and 26  Similar to lines 20, 21 and 22.
Line 27
This line announces the completion of the compare for the FMT FOSC1A.
Chapter 14. Troubleshooting

The following topics provide you with technical references to help you troubleshoot and diagnose IMS Library Integrity Utilities problems.

Topics:
- “IMS Library Integrity Utilities return codes” on page 438
- “IMS Library Integrity Utilities abend codes” on page 444
- “IMS messages” on page 445
- “IMS Library Integrity Utilities messages” on page 446
- “How to look up message explanations” on page 521
- “Gathering diagnostic information” on page 522
- “Diagnostics Aid” on page 523
IMS Library Integrity Utilities return codes

IMS Library Integrity Utilities generates return codes to indicate the success or failure of a job.

The following topics describe how to read the return codes of each utility.

Integrity Checker and LICON utility return codes

This reference topic explains the return codes of the Integrity Checker utility and the LICON utility.

Because the DMB verification routine does not return directly to the OS dispatcher, there is no job step return code that is directly related to the DMB verification results.

The LICON utility returns four return codes as summarized in the following table.

Table 24. LICON utility return codes

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion.</td>
</tr>
<tr>
<td></td>
<td>The program ended successfully.</td>
</tr>
<tr>
<td>4</td>
<td>Warning message.</td>
</tr>
<tr>
<td></td>
<td>Warning messages were issued, but the requested operation was completed.</td>
</tr>
<tr>
<td>8</td>
<td>Error message.</td>
</tr>
<tr>
<td></td>
<td>Error messages were issued, and requested operations for some (but not all) databases failed.</td>
</tr>
<tr>
<td>16</td>
<td>Error message.</td>
</tr>
<tr>
<td></td>
<td>Error messages were issued.</td>
</tr>
<tr>
<td></td>
<td>• No databases or RDEs were successfully processed.</td>
</tr>
<tr>
<td></td>
<td>• A severe error occurred.</td>
</tr>
<tr>
<td></td>
<td>The LICON utility ended immediately after the error condition detected. It skips processing any subsequent commands.</td>
</tr>
</tbody>
</table>

Consistency Checker return codes

This reference topic explains the return codes of the Consistency Checker utility.

Consistency Checker returns four return codes as summarized in the following table.

Table 25. Consistency Checker utility return codes

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion.</td>
</tr>
<tr>
<td></td>
<td>The program ended successfully.</td>
</tr>
<tr>
<td></td>
<td>If, for any reason, reports are not produced, there is an explanation in the activity log.</td>
</tr>
<tr>
<td>4</td>
<td>Warning message.</td>
</tr>
<tr>
<td></td>
<td>Warning messages were issued, but the requested operation was completed.</td>
</tr>
</tbody>
</table>
Multiple Resource Checker return codes

This reference topic explains the return codes of the Multiple Resource Checker utility.

Multiple Resource Checker returns three return codes as summarized in the following table.

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion. The program ended successfully. If, for any reason, reports are not produced, there is an explanation in the activity log.</td>
</tr>
<tr>
<td>4</td>
<td>Warning message. Warning messages were issued, but the requested operation was completed.</td>
</tr>
<tr>
<td>8</td>
<td>Error message. Error messages were issued, and some requested operations were skipped or ended unsuccessfully because severe errors were detected.</td>
</tr>
</tbody>
</table>

DBD/PSB/ACB Compare, Mapper, and Reversal return codes

This reference topic explains the return codes of the DBD/PSB/ACB Compare utility, the DBD/PSB/ACB Mapper utility, the DBD/PSB/ACB Reversal utility, and the Reversal Site Default Generation utility.

The DBD/PSB/ACB Compare, Mapper, Reversal, and Reversal Site Default Generation utilities return three return codes as summarized in the following table.

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion. The program was successfully executed. If, for any reason, maps and reports are not produced, there is an explanation in the activity log.</td>
</tr>
</tbody>
</table>
Table 27. DBD/PSB/ACB Compare, Mapper, and Reversal utility return codes (continued)

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Warning message. Warning messages were issued, but the requested operation was completed.</td>
</tr>
<tr>
<td>8</td>
<td>Unsuccessful execution. Some requested operations were skipped or executed unsuccessfully because severe errors were detected.</td>
</tr>
<tr>
<td>16</td>
<td>Error message. Error messages were issued. The requested operations were not performed because the control statement errors were detected.</td>
</tr>
</tbody>
</table>

Catalog Manager return codes

This reference topic explains the return codes of the Catalog Manager utility.

The Catalog Manager utility returns one of the following three return codes.

Table 28. Catalog Manager utility return codes

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion. The program ended successfully.</td>
</tr>
<tr>
<td>4</td>
<td>Warning message. Warning messages were issued, but the requested operation was completed.</td>
</tr>
<tr>
<td>8</td>
<td>Error message. Error messages were issued, and some requested operations were skipped or ended unsuccessfully because severe errors were detected.</td>
</tr>
</tbody>
</table>

Advanced ACB Generator return codes

This reference topic explains the return codes of the Advanced ACB Generator utility.

Whenever an error condition is detected, an error message is issued and a return code is set. Advanced ACBGEN utility displays both IMS generated DFSnnnn messages and its own FABQnnnn messages. It also honors and reports any return code set in conjunction with any DFSnnnn messages. Continuation of the ACBGEN process depends upon the error condition. The highest return code encountered is the return code that is passed back to the MVS job step termination routine and displayed as the job step completion code. It can be tested by including a COND=operand in the EXEC JCL statement of a later job step.

MFS Reversal return codes

This reference topic explains the return codes of the MFS Reversal utility.

The return codes of the MFS Reversal utility are summarized in the following table.
Table 29. MFS Reversal utility return codes

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion. The program ended successfully.</td>
</tr>
<tr>
<td>4</td>
<td>Warning message. Warning messages were issued, but the requested operation was completed.</td>
</tr>
<tr>
<td>8 or higher</td>
<td>Error message. Error messages were issued. The program ended unsuccessfully.</td>
</tr>
</tbody>
</table>

MFS Compare return codes

This reference topic explains the return codes of the MFS Compare utility.

The return codes of the MFS Compare utility are as summarized in the following table.

Table 30. MFS Compare utility return codes

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion. The program ended successfully.</td>
</tr>
</tbody>
</table>
| 8 or higher | Either of the following errors occurred: 
  • The program found differences between MFS control blocks. 
  • Error messages were issued. The program ended unsuccessfully. |

IMS Library Integrity Utilities return codes under IMS Administration Tool

This reference topic explains the return codes of IMS Library Integrity Utilities when the utility is used under IMS Administration Tool.

IMS Library Integrity Utilities return codes under IMS Administration Tool

When IMS Library Integrity Utilities is used under IMS Administration Tool, IMS Library Integrity Utilities returns one of the return codes summarized in the following table.

Table 31. IMS Library Integrity Utilities return codes under IMS Administration Tool

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion. The program ended successfully.</td>
</tr>
</tbody>
</table>
Table 31. IMS Library Integrity Utilities return codes under IMS Administration Tool (continued)

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Warning message. Warning messages were issued while the utility decoded the DBD or PSB source, but the requested operation completed. The warning messages are printed in the comment lines in the DBD or PSB source code.</td>
</tr>
<tr>
<td>4</td>
<td>Warning message. Warning messages were issued, but the requested operation completed. • One or more differences or mismatches were detected during the compare operation. • The specified PSB is not valid for IMS SQL. • The DBD or PSB resource was not found. • DBD instances with an old DB Version are not used because database versioning is not enabled.</td>
</tr>
<tr>
<td>8</td>
<td>Error message. Error messages were issued, and the requested operation failed.</td>
</tr>
<tr>
<td>12</td>
<td>Error message. Error messages were issued, and the requested operation failed. DBD library, PSB library, ACB library, or IMS catalog was not discovered.</td>
</tr>
<tr>
<td>16</td>
<td>Error message. Error messages were issued, and the requested operation failed. IMS ID is not registered.</td>
</tr>
<tr>
<td>99</td>
<td>Error message. Error messages were issued, and the requested operation failed. Unexpected error occurred.</td>
</tr>
</tbody>
</table>

FABXAEXP return codes (Export function)

The following table summarizes the return codes of the FABXAEXP program. This program is invoked by the JCL that is generated by the Catalog or ACBLIB export function of IMS Administration Tool.

Table 32. FABXAEXP return codes

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion. The program ended successfully.</td>
</tr>
</tbody>
</table>
### Table 32. FABXAEXP return codes (continued)

<table>
<thead>
<tr>
<th>Return code</th>
<th>Meaning</th>
</tr>
</thead>
</table>
| 2           | Warning message.  
  Warning messages were issued while decoding one or more DBDs, PSBs, or both, but the requested operation completed.  
  The warning messages are printed in the comment lines in the DBD or PSB sources. |
| 4           | Warning message.  
  Warning messages were issued, but the requested operation completed.  
  One or more DBD or PSB resources were not found. |
| 8           | Error message.  
  Error messages were issued, and the requested operation failed. |
| 12          | Error message.  
  Error messages were issued, and the requested operation failed.  
  DBD library, PSB library, ACB library, or IMS Catalog was not discovered. |
| 16          | Error message.  
  Error messages were issued, and the requested operation failed.  
  IMS ID is not registered. |
| 99          | Error message.  
  Error messages were issued, and the requested operation failed.  
  Unexpected error occurred. |
IMS Library Integrity Utilities abend codes

IMS Library Integrity Utilities issues abend codes when a utility terminates abnormally.

The following reference topics provide detailed information about IMS Library Integrity Utilities abend codes. Use this information to help you with troubleshooting.

Integrity Checker abend codes
Integrity Checker uses abend codes U3xxx. Before issuing the abend code, Integrity Checker always writes a message identifying the problem. The message number is the same as the abend code. If no message with the same number is found, check for error messages that describe the error conditions.

Consistency Checker abend codes
Consistency Checker uses abend codes U2030, U2032, U2035, U2042, U2045, U2050, U2051, and U2052. Before issuing any of these abend codes, Consistency Checker always writes a message identifying the problem. The message number is the same as the abend code.

Multiple Resource Checker abend codes
Multiple Resource Checker uses abend codes U3xxx. Before issuing the abend code, Multiple Resource Checker always writes a message identifying the problem. The message number is the same as the abend code.

DBD/PSB/ACB Compare abend codes
DBD/PSB/ACB Compare uses abend codes U0008, U0014, U0015, and U0016. Before issuing any of these abend codes, DBD/PSB/ACB Compare always writes a message identifying the problem. The message number is the same as the abend code.

DBD/PSB/ACB Mapper abend codes
The DBD/PSB/ACB Mapper utility uses abend codes U0021, U0022, U0026, U0027, and U0028. Before issuing any of these abend codes, DBD/PSB/ACB Mapper always writes a message identifying the problem. The message number is the same as the abend code.

DBD/PSB/ACB Reversal abend codes
The DBD/PSB/ACB Reversal utility uses abend codes U0001, U0002, U0003, U0004, and U0005. The Reversal Site Default Generation utility uses abend codes U3001, U3002, and U3003. Before issuing any of these abend codes, the DBD/PSB/ACB Reversal utility or the Reversal Site Default Generation utility always writes a message identifying the problem. The message number is the same as the abend code.

MFS Reversal abend codes
The MFS Reversal utility uses abend codes U3xxx. Before issuing the abend code, the MFS Reversal utility always writes a message that identifies the problem. The message number is the same as the abend code.
**IMS messages**

Advanced ACB Generator displays both IMS generated DFSnnnn messages and its own FABQnnnn messages.

The meanings of the DFSnnnn messages have not been changed; see *IMS Messages and Codes* for the meaning of the IMS messages. Certain error conditions might also cause an abend to be issued. These abend codes are also documented in the *IMS Messages and Codes*. 
IMS Library Integrity Utilities messages

Use the information in these messages to help you diagnose and solve IMS Library Integrity Utilities problems.

IMS Library Integrity Utilities messages adhere to the following format:

FABxnnnny

Where:

<table>
<thead>
<tr>
<th>FABx</th>
<th>Indicates that the message was issued by IMS Library Integrity Utilities. x is one of L, M, N, Q, V, W, and X.</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>Indicates that the message was issued by the Integrity Checker utility, the LICON utility, the Consistency Checker utility, or the DBD/PSB/ACB Compare utility.</td>
</tr>
<tr>
<td>M</td>
<td>Indicates that the message was issued by the DBD/PSB/ACB Mapper utility.</td>
</tr>
<tr>
<td>N</td>
<td>Indicates that the message was issued by the DBD/PSB/ACB Reversal utility or the Reversal Site Default Generation utility.</td>
</tr>
<tr>
<td>Q</td>
<td>Indicates that the message was issued by the Advanced ACBGEN utility.</td>
</tr>
<tr>
<td>V</td>
<td>Indicates that the message was issued by the MFS Reversal utility or the MFS Compare utility.</td>
</tr>
<tr>
<td>W</td>
<td>Indicates that the message was issued by the Multiple Resource Checker utility.</td>
</tr>
<tr>
<td>X</td>
<td>Indicates that the message was issued by the Catalog Manager utility or issued while using the DBD/PSB Map Viewer.</td>
</tr>
</tbody>
</table>

nnn   Indicates the message identification number.

y     Indicates the severity of the message.

E     Indicates that an error occurred, which might or might not require operator intervention.

I     Indicates that the message is informational only.

W     Indicates that the message is a warning to alert you to a possible error condition.

Each message also includes the following information:

**Explanation:**
The Explanation section explains what the message text means, why it occurred, and what its variables represent.

**System action:**
The System action section explains what the system will do in response to the event that triggered this message.

**User response:**
The User response section describes whether a response is necessary, what the appropriate response is, and how the response will affect the system or program.

**Module:**
The Module section indicates which module or modules are affected.
FABL messages

Messages that are issued by the Integrity Checker utility, the LICON utility, the Consistency Checker utility, and the DBD/PSB/ACB Compare utility begin with the prefix FABL.

FABL0001I CONTROL CARD SUPPLIED IS: echo of control statement

Explanation: This message is the echo of the SYSIN control statements that are checked by this utility.

System action: The DBD/PSB/ACB Compare utility continues processing.

User response: None. This message is informational.

FABL0002W INVALID STATEMENT IN SYSIN DATASET

Explanation: A control statement with invalid format was found in the SYSIN data set.

System action: The DBD/PSB/ACB Compare utility skips this control statement and continues processing.

User response: Correct the format of this control statement, and rerun the job.

FABL0003E LIBRARY MISSING FOR [DBD | PSB | ACB] COMPARE

Explanation: The specification of the libraries required to execute the function is missing or invalid.

System action: Skips the reporting process for this function.

User response: Determine whether the required libraries are specified in DD statements. Correct the DD statements for load module libraries, and rerun the job.

FABL0004I [DBD | PSB | ACB] TO BE PROCESSED [IS member | ARE members]

Explanation: This message shows the name of the member or the members that are to be processed. Only one member name is printed when the names of the members that are specified in the control statement are the same.

System action: The DBD/PSB/ACB Compare utility continues processing.

User response: None. This message is informational.

FABL0005W NO MEMBER FOUND FOR member IN [DBDLIB | DBDLIB2 | PSBLIB | PSBLIB2 | ACBLIB | ACBLIB2]

Explanation: The specified member, or one or more of the members specified by a wildcard, were not found in the DBD/PSB/ACB library. The member is the specified member name.

System action: The DBD/PSB/ACB Compare utility continues processing, not printing the report of the member.

User response: Determine whether the member is correct. If it is incorrect, search the library that has the member. Correct the problem, and rerun the job.

FABL0006I NO DIFFERENCE FOUND DURING COMPARE [DBD | PSB | ACB] = members

Explanation: The Compare function ran normally, and no difference was found between the members named members in the specified libraries. Only one member name is printed when the names of the members that are specified in the control statement are the same.

System action: The DBD/PSB/ACB Compare utility continues processing.

User response: None. This message is informational.

FABL0007W DIFFERENCE FOUND DURING COMPARE [DBD | PSB | ACB] = members

Explanation: The Compare function ran normally, and a difference was found between the members named members in the specified libraries. Only one member name is printed when the names of the members that are specified in the control statement are the same.

System action: The DBD/PSB/ACB Compare utility generates a compare report and continues processing.

User response: None.

FABL0008E GETMAIN FAILED

Explanation: The program could not obtain enough area with the GETMAIN macro.

System action: The DBD/PSB/ACB Compare utility ends abnormally.

User response: If the region size specified is too small, increase the REGION size in the JOB statement in the JCL, and rerun the utility.

FABL0009E ERROR READING ACB=member IN acblib_ddname

Explanation: An error occurred in the reading of an ACB member in the acblib_ddname.

System action: The DBD/PSB/ACB Compare utility
continues processing without reporting this ACB member.

User response: Determine the cause of failure, correct it, and rerun the utility.

FABL0010E  BLDL FAILED FOR ACB DIRECTORY
            IN acblib_ddname

Explanation: An error occurred while a BLDL macro was being issued for the acblib_ddname.

System action: The DBD/PSB/ACB Compare utility skips the reporting process for ACB compare.

User response: Determine the cause of the BLDL macro failure, correct it, and rerun the utility.

FABL0011E  UNSUPPORTED VERSION, ACB=xxxx

Explanation: The ACB member generated by IMS version xxxx is not supported.

System action: The DBD/PSB/ACB Compare utility continues processing without reporting for this ACB member.

User response: Check the ACB member and the ACB library version.

FABL0012W  MAXIMUM SYSIN CARDS EXCEEDED

Explanation: The DBD/PSB/ACB Compare utility supports a maximum of 9999 control statements.

System action: Processes the first 9999 statements and ignores the rest.

User response: Rerun the ignored cards.

FABL0013E  MEMBER TYPE IS DIFFERENT,
            ACB=members

Explanation: The DBD/PSB/ACB Compare utility detected a difference in the ACB member type: one is a
            PSB-type ACB and the other is not. Only one member name is printed when the names of the members that are
            specified in the control statement are the same.

System action: The DBD/PSB/ACB Compare utility continues processing without creating a compare report
            for these members.

User response: None.

FABL0014E  SYSOUT DID NOT OPEN

Explanation: The SYSOUT data set could not be opened during initialization.

System action: The DBD/PSB/ACB Compare utility ends abnormally.

User response: Determine the cause of the failure.

FABL0015E  SYSIN DID NOT OPEN

Explanation: The SYSIN data set could not be opened during initialization.

System action: The DBD/PSB/ACB Compare utility ends abnormally.

User response: Determine the cause of the failure.

FABL0016E  SYSPRINT DID NOT OPEN

Explanation: The SYSPRINT data set could not be opened during initialization.

System action: The DBD/PSB/ACB Compare utility ends abnormally.

User response: Determine the cause of the failure.

FABL0017E  MEMBER TYPE IS DIFFERENT,
            ACB=members (ACB1=member_type,
            ACB2=member_type)

Explanation: The utility detected a difference in the ACB member type. member_type is one of the types:
            DEB, MSDB, or NOT FP. Only one member name is printed when the names of the members that are specified in the control statement are the same.

System action: The DBD/PSB/ACB Compare utility continues processing without creating a compare report for these members.

User response: None.

FABL0018E  ERROR LOADING [DBD | PSB]
            NAMED member IN library_ddname
            (ABEND CODE=abend_code REASON
            CODE=reason_code)

Explanation: An error occurred while loading
            DBD/PSB member in library_ddname. abend_code is the
            system abend code, and reason_code is the reason code.

System action: Skips this member and tries to load the next member if it exists.

User response: Determine the cause of the load error. Correct the problem, and rerun the utility.

FABL0019W  [DBD | PSB] member IS NOT A VALID
            [DBD | PSB] IN library_ddname. ERROR
            IS DETECTED IN control_block_name

Explanation: DBD/PSB control_block_name in
            library_ddname was loaded, but was found not to be valid. If the invalid block can be identified, the block
            name follows.

System action: Skips this member and tries to load the next member if there is one.

User response: Determine whether the member is a DBD or a PSB. If the member is a DBD or a PSB,
regenerate it. If it is not, ignore this message.

FABL0020E VERSION NOT MATCHED, ACB1=x.x
ACB2=y.y

Explanation: The ACB in ACBLIB is generated by IMS version x.x, and the ACB in ACBLIB2 is generated by IMS version y.y. Block-level compare does not support comparing ACB members of different IMS releases.

System action: The DBD/PSB/ACB Compare utility continues processing without reporting for these ACB members.

User response: Check the ACB members and the ACB library version. If you want to compare the members of different IMS releases, generate the source-level compare report.

FABL0021W INVALID PARAMETER parameter IN [NOCOMP | REPORT] STATEMENT

Explanation: An incorrect parameter parameter was found in the NOCOMP or the REPORT statement of SYSIN data set.

System action: The DBD/PSB/ACB Compare utility skips this incorrect parameter and continues processing.

User response: Correct this parameter of the NOCOMP or the REPORT statement, and rerun the job.

FABL0022I COMPARE MODE IS [NOCOMP | REPORT] =mode

Explanation: The DBD/PSB/ACB Compare utility proceeds with NOCOMP mode mode or REPORT mode mode.

System action: Processing continues.

User response: None. This message is informational.

FABL0023I [MEMBER member | MEMBERS members] PROCESSED

Explanation: This message shows the name of the member or the members that were processed. When the names of the members are the same, the member name is printed only once.

System action: The DBD/PSB/ACB Compare utility continues processing.

User response: None. This message is informational.

FABL0024E READ ERROR ON [DBD | PSB | ACB] DIRECTORY

Explanation: A read error occurred in the reading of the directory.

System action: The DBD/PSB/ACB Compare utility skips this function and continues processing. However, return code 8 is issued.

User response: None.

FABL0025W NO DATA IN SYSIN

Explanation: No control statement is specified in the SYSIN data set.

System action: The DBD/PSB/ACB Compare utility ends without compare.

User response: Specify the control statement in the SYSIN data set, and rerun the job.

FABL0026W NO MEMBER NAME IS SPECIFIED description

Explanation: No member name is specified in the control statement. If you have specified a colon in the control statement, you must specify two member names in member1:member2 format. When a colon is used in the control statement, description indicates the missing member.

System action: The DBD/PSB/ACB Compare utility skips this control statement and continues processing.

User response: Specify the member name in the control statement, and rerun the job.

FABL0027W INVALID MEMBER NAME IS SPECIFIED description

Explanation: An invalid member name was specified in the control statement. For example, a member name containing more than eight characters is specified. When different member names are specified in the member1:member2 format and one of the member names is an invalid member name, description indicates the invalid member.

System action: The DBD/PSB/ACB Compare utility skips this control statement and continues processing.

User response: Specify the correct member name in the control statement, and rerun the job.

FABL0028E INCOMPATIBLE CONTROL BLOCK STRUCTURE, ACB=members
VERSION=x.x SECTION=section

Explanation: The DBD/PSB/ACB Compare function failed, because the control block structures of the members are incompatible. Either member must be generated by the latest maintenance level of IMS version x.x. section is the section name (control block name).

System action: The DBD/PSB/ACB Compare utility continues processing without reporting these ACB members.

User response: Apply the latest PTF, and rerun the utility.
FABL0029E DECODE [DBD | PSB] PROCESSING FAILED WITH RC=yy

Explanation: The Reversal utility returned a nonzero return code. The return code is yy.

System action: The DBD/PSB/ACB Compare utility ends with a return code of 08.

User response: Check the preceding messages that explain the error conditions. Correct the error, and rerun the job.

FABL0030E UNSUPPORTED IMS VERSION (x.x) DETECTED FOR ACB SOURCE COMPARE. MEMBER: member ACBLIB: [ACBLIB | ACBLIB2]

Explanation: The ACB member generated by IMS Version x.x is not supported for ACB Source Compare.

System action: The DBD/PSB/ACB Compare function ends with a return code of 08.

User response: Check the ACB member and the ACB library version.

FABL0031W Unsupported DBD member IN [DBDLIB | DBDLIB2]

Explanation: The specified DBD member in the DBD library is not supported for the DBD/PSB/ACB Compare utility.

System action: The DBD/PSB/ACB Compare utility skips this member and continues processing.

User response: Check the DBD member.

FABL0032I DBD dbname FOR GSAM OR LOGICAL IS NOT COMPARED

Explanation: The DBD/PSB/ACB Compare utility does not compare the GSAM or the logical DBD because there is no ACB for a GSAM or a logical database. dbname is the DBD member name that was specified.

System action: The DBD/PSB/ACB Compare utility continues processing.

User response: None. This message is informational.

FABL0033I GSAM PCB (NUM=xxx) IN PSB psbname IS NOT COMPARED

Explanation: The DBD/PSB/ACB Compare utility does not compare the GSAM PCBs because the ACB contains no information about GSAM PCBs. psbname is the PSB member name that was specified, and xxx is the number of GSAM PCBs in the PSB that was specified.

System action: The DBD/PSB/ACB Compare utility continues processing.

User response: None. This message is informational.

FABL0034W INCORRECT OPTION IN SYSIN

Explanation: There is an incorrect option in the SYSIN data set.

System action: The DBD/PSB/ACB Compare utility skips this control statement and continues processing.

User response: Correct the option, and rerun the job.

FABL0035W acbname WAS NOT [DBD | PSB] TYPE ACB MEMBER

Explanation: The specified member was not a DBD-type ACB member. The specified acbname is the member name that was specified in the SYSIN card.

System action: The DBD/PSB/ACB Compare utility skips this control statement and continues processing.

User response: None.

FABL0036W NO [DBD | PSB] TYPE ACB MEMBER FOUND FOR acbname IN ACBLIB

Explanation: The DBD-type ACB members or the PSB-type ACB members that were specified by a wildcard, were not found in the ACB library. The acbname is the specified member name.

System action: The DBD/PSB/ACB Compare utility skips this control statement and continues processing.

User response: None.

FABL0037E ddname DID NOT OPEN

Explanation: The data set ddname could not be opened.

System action: The DBD/PSB/ACB Compare utility skips the process related to this DD statement.

User response: Determine the cause of the open failure. Correct the error, and rerun the utility.

FABL0038W WILD CARD CHARACTERS CANNOT BE USED TO SPECIFY THE SECOND MEMBER NAME. MEMBER NAMES ARE member1member2

Explanation: Wildcard characters cannot be used to describe member2.

System action: The DBD/PSB/ACB Compare utility skips the control statement that contains the invalid member name and continues processing.

User response: None.
FABL0039W  SECOND MEMBER NAME CANNOT BE SPECIFIED WHEN COMPARING DIFFERENT TYPES OF CONTROL BLOCKS.

Explanation: The second member name can be specified only when comparing control blocks that have different names but that are of the same type. The DBD/PSB/ACB Compare utility cannot compare control blocks that have different names and that are of different types.

System action: The DBD/PSB/ACB Compare utility skips this control statement that contains the invalid member name and continues processing.

User response: None.

FABL0040W  RDMVTAB CSECT IS CUSTOMIZED: MEMBER=member IN library_ddname.

Explanation: While processing member in library_ddname, the DBD/PSB/ACB Compare utility detected one or more customized fields in RDMVTAB CSECT (described by the DMBDACS DSECT) that contain the randomizing information. One or more of the following fields are detected as customized by the DBD/PSB/ACB Compare utility.

<table>
<thead>
<tr>
<th>Detectable DBD type</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD and DB TYPE ACB</td>
<td>DMBDASZE</td>
<td>The size of RDMVTAB CSECT</td>
</tr>
<tr>
<td>DBD and DB TYPE ACB</td>
<td>DMBDAKL</td>
<td>The executable key length of root</td>
</tr>
<tr>
<td>DBD</td>
<td>DMBDANME</td>
<td>The name of randomizer module</td>
</tr>
<tr>
<td>DBD</td>
<td>DMBDARAP</td>
<td>The number of root anchor points or blocks</td>
</tr>
<tr>
<td>DBD</td>
<td>DMBDABLK</td>
<td>The number of the highest blocks that are directly addressed</td>
</tr>
<tr>
<td>DBD</td>
<td>DMBDABYM</td>
<td>The maximum number of bytes</td>
</tr>
</tbody>
</table>

If the utility processes a block-level compare and detects one or more customized fields in RDMVTAB CSECT, only the detectable fields are compared, and other fields are not compared.

If the utility processes a source-level compare, all of the fields in RDMVTAB CSECT are not compared regardless of whether the RDMVTAB CSECT is customized.

System action: The DBD/PSB/ACB Compare utility continues processing.

User response: None.

FABL0041I  COMPAIRED number_of_control_blocks1 resource1 WITH resource2. DETECTED number_of_control_blocks2 IDENTICAL CONTROL BLOCKS AND number_of_control_blocks3 MISMATCHED CONTROL BLOCKS.

Explanation: This informational message summarizes the results of a compare operation that was done by the DBD/PSB/ACB Compare utility. A FABL0041I message is issued for each type of control block; therefore, multiple FABL0041I messages might be issued.

• resource1 and resource2 show the type of the compared members, which is DBD, PSB, or ACB.
• number_of_control_blocks1 shows the total number of DBDs, PSBs, or ACBs that were compared.
• number_of_control_blocks2 shows the number of DBDs, PSBs, or ACBs that are identical.
• number_of_control_blocks3 shows the number of DBDs, PSBs, or ACBs that are different.

For example, assume that the DBD/PSB/ACB Compare utility compares the following control statements, and results:

<table>
<thead>
<tr>
<th>Control statement</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD=DBD00001A:DBD0002A</td>
<td>Different</td>
</tr>
<tr>
<td>DBD=DBD00003A</td>
<td>Match</td>
</tr>
<tr>
<td>PSB=PSB00001A:PSB0002A</td>
<td>Different</td>
</tr>
<tr>
<td>ACB=ACB00001A</td>
<td>Different</td>
</tr>
</tbody>
</table>

In this case, the following FABL0041I messages are issued:

• COMPARED 2 DBD WITH DBD. DETECTED 1 IDENTICAL CONTROL BLOCKS AND 1 MISMATCHED CONTROL BLOCKS.
• COMPARED 1 PSB WITH PSB. DETECTED 0 IDENTICAL CONTROL BLOCKS AND 1 MISMATCHED CONTROL BLOCKS.
• COMPARED 1 ACB WITH ACB. DETECTED 0 IDENTICAL CONTROL BLOCKS AND 1 MISMATCHED CONTROL BLOCKS.

System action: The DBD/PSB/ACB Compare utility continues processing.

User response: None. This message is informational.

FABL0042I  COMPAIRED number_of_source1 resource1 WITH resource2. DETECTED number_of_source2 IDENTICAL SOURCES AND number_of_source3 MISMATCHED SOURCES.

Explanation: This informational message summarizes the results of a compare operation that was done by the DBD/PSB/ACB Compare utility. A FABL0042I message is issued for each type of comparison; therefore, multiple FABL0042I messages might be issued.
• resource1 and resource2 show the type of the compared members, which is DBD, PSB, or ACB.
• number_of_resource1 shows the total number of DBDs, PSBs, or ACBs that were compared at their source levels.
• number_of_resource2 shows the number of DBDs, PSBs, or ACBs that are identical at their source levels.
• number_of_resource3 shows the number of DBDs, PSBs, or ACBs that are different at their source levels.

For example, assume that the DBD/PSB/ACB Compare utility compared the following control statements, and results:

<table>
<thead>
<tr>
<th>Control statement with REPORT=SOURCE</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD=DBD0001A:DBD0002A</td>
<td>Different</td>
</tr>
<tr>
<td>DBD=DBD0003A</td>
<td>Match</td>
</tr>
<tr>
<td>PSB=PSB0001A:PSB0002A</td>
<td>Different</td>
</tr>
<tr>
<td>ACB=DBD0002A</td>
<td>Different</td>
</tr>
<tr>
<td>DBD=DBD0001A,ACB</td>
<td>Different</td>
</tr>
<tr>
<td>PSB=PSB0001A,ACB</td>
<td>Match</td>
</tr>
<tr>
<td>ACB=DBD0001A,DBD</td>
<td>Different</td>
</tr>
<tr>
<td>ACB=PSB0001A,PSB</td>
<td>Match</td>
</tr>
</tbody>
</table>

In this case, the following FABL0042I messages are issued:

• COMPARED 2 DBD WITH DBD. DETECTED 1 IDENTICAL SOURCES AND 1 MISMATCHED SOURCES.
• COMPARED 1 PSB WITH PSB. DETECTED 0 IDENTICAL SOURCES AND 1 MISMATCHED SOURCES.
• COMPARED 1 ACB WITH ACB. DETECTED 0 IDENTICAL SOURCES AND 1 MISMATCHED SOURCES.
• COMPARED 1 DBD WITH ACB. DETECTED 0 IDENTICAL SOURCES AND 1 MISMATCHED SOURCES.
• COMPARED 1 PSB WITH ACB. DETECTED 1 IDENTICAL SOURCES AND 0 MISMATCHED SOURCES.
• COMPARED 1 ACB WITH DBD. DETECTED 0 IDENTICAL SOURCES AND 1 MISMATCHED SOURCES.
• COMPARED 1 ACB WITH PSB. DETECTED 1 IDENTICAL SOURCES AND 0 MISMATCHED SOURCES.

System action: The DBD/PSB/ACB Compare utility continues processing.

User response: None. This message is informational.
**FABL0047W**  GSAM DBD `dbname` IS NOT COMPARED

**Explanation:** IMS Library Integrity Utilities does not compare the indicated GSAM DBD because it found no ACBs for the GSAM database in the ACB library.

**dbname** is the DBD member name that was specified.

**System action:** IMS Library Integrity Utilities continues processing.

**User response:** None. This message is informational.

---

**FABL0047W**  VERSION AND EXIT PARAMETERS ARE NOT COMPARED FOR DBD `dbname`. NO PSB REFERS TO THIS DBD IN `resource`

**Explanation:** The following parameters are not compared because these parameters could not be obtained from the PSB member that refers the reported DBD member.

- The VERSION parameter of the DBD statement
- The EXIT parameter of the DBD and SEGM statements

**System action:** The DBD/PSB/ACB Compare utility continues processing.

**User response:** Check the DBD and PSB members to generate the expected result.

---

**FABL0048E**  ACCESS FAILED FOR `cataloghlq`.  

**Explanation:** IMS Library Integrity Utilities detected an error while accessing the IMS catalog directory.

**System action:** IMS Library Integrity Utilities ends with a return code of 8.

**User response:** None. This message is informational.

---

**FABL0049I**  COMARED number_of_member1 MEMBERS FOUND IN `resource1` AND IMS CATALOG. DETECTED number_of_member2 IDENTICAL MEMBERS AND number_of_member3 MISMATCHED MEMBERS.

**Explanation:** This informational message summarizes the results of a compare operation that was done by IMS Library Integrity Utilities. A FABL0049I message is issued for each type of control block; therefore, multiple FABL0049I messages might be issued.

- `resource1` show the type of the compared members, which is ACB.
- `number_of_member1` shows the total number of members that were compared.
- `number_of_member2` shows the number of members that are identical.
- `number_of_member3` shows the number of members that are different.

**System action:** IMS Library Integrity Utilities continues processing.

**User response:** None. This message is informational.

---

**FABL0050E**  NAME/TOKEN SERVICE `service` FAILED. NAME: `nametoken`. RC=`rc`

**Explanation:** The process failed in the z/OS MVS Name/Token Service. `service` shows the service name. `rc` is the return code from the Name/Token service.

**System action:** IMS Library Integrity Utilities ends with a user abend code of U0050.

**User response:** Find the cause of the error. For the return code, see the z/OS MVS Programming: Assembler Services Reference, Volume 2 (IARR2V-XCTLX). If necessary, correct the warning condition and rerun the job.

---

**FABL0101I**  LIU INTEGRITY CHECKER NOW ACTIVE WITH LICON: `dsn`

**Explanation:** The Integrity Checker utility is now active with the LICON data set `dsn`.

**System action:** Processing continues.

**User response:** None. This message is informational.

---

**FABL0102I**  LIU INTEGRITY CHECKER INITIALIZATION COMPLETED

**Explanation:** The Integrity Checker utility has been initialized.

**System action:** Processing continues.

**User response:** None. This message is informational.

---

**FABL0104E**  LIU INTEGRITY CHECKER INITIALIZATION FAILED

**Explanation:** The initialization of the Integrity Checker utility failed with some error conditions.

**System action:** The IMS online subsystem or the IMS batch region ends abnormally.

**User response:** Check the preceding messages that explain the error conditions. Correct the error conditions. Restart the IMS online subsystem or rerun the batch job.

---

Chapter 14. Troubleshooting  453
FABL0105I • FABL0202W

FABL0105I  LIU INTEGRITY CHECKER TERMINATION COMPLETED

Explanation: The Integrity Checker utility has been terminated.

System action: Processing continues. The IMS online region or the IMS batch region ends normally.

User response: None. This message is informational.

FABL0107E  LIU INTEGRITY CHECKER NOW INACTIVE. RSN=ssss

Explanation: The Integrity Checker processing ended with error condition ssss.

System action: The IMS online region continues processing without Integrity Checker.

User response: Check the preceding messages that explain the error conditions. Correct the error conditions and restart the IMS online subsystem. If no preceding message is found, contact IBM Software Support.

FABL0108E  PUT FAILED FOR DDNAME: ddname

Explanation: The PUT macro that was issued to the data set whose DD name is ddname has failed.

System action: Processing continues without using the indicated data set.

User response: Check the status of data set ddname. Correct the error conditions and restart the IMS online subsystem.

FABL0109E  LOAD FAILED FOR MODULE module

Explanation: The LOAD macro failed. module is either FABLRTS0 or FABLRTRx where x is 6, 7, 8, 9, A, B, C, D, E, or F.

System action: The job requesting database authorization ends abnormally.

User response: Check that the STEPLIB concatenation of the job or the DBRC cataloged procedure contains the correct load module library that contains the Integrity Checker load modules. Correct the error conditions. Restart the IMS online subsystem or rerun the job.

FABL0110W  LIU INTEGRITY CHECKER STOPPED PROCESSING

Explanation: The Integrity Checker utility could not successfully complete its initialization. The Integrity Checker utility processing was stopped because INITERR=W was specified in the global option module.

System action: The IMS online subsystem or the IMS batch region continues processing without the Integrity Checker function.

User response: Check the preceding messages that explain the error conditions. If necessary, correct the error conditions and restart the IMS online subsystem or rerun the batch job.

FABL0111E  UNSUPPORTED LEVEL OF IMS IS BEING USED

Explanation: The Integrity Checker utility is run under an unsupported version of IMS.

System action: The IMS online subsystem or the IMS batch region ends abnormally.

User response: Correct the error conditions. Restart the IMS online subsystem or rerun the batch job.

FABL0112W  UNSUPPORTED LEVEL OF IMS IS BEING USED: nn.nn

Explanation: The Integrity Checker utility was run under an unsupported version of IMS. nn.nn shows the version and release of IMS that is being used.

System action: Processing continues; the database is not verified.

User response: Determine if the version of IMS is correct. Correct the error conditions and rerun the batch job.

FABL0114I  LIU INTEGRITY CHECKER ACTIVATED. IMS VERSION IS version

Explanation: The Integrity Checker utility activated the DMB verification process. This message also indicates the version of IMS that is being used.

System action: Processing continues.

User response: None. This message is informational.

FABL0201I  RDE CREATED FOR DBD: dddddddd [AREA: aaaaaaaaa]

Explanation: The registered DMB entry (RDE) has been successfully created for non-HALDB full-function database or HALDB partition dddddddd, or area aaaaaaaaa of database dddddddd.

System action: Processing continues.

User response: None. This message is informational.

FABL0202W  SIZE OF DMB INFORMATION HAS EXCEEDED MAXIMUM RECORD LENGTH OF RDE

Explanation: The size of the DMB information has exceeded the maximum record length of a registered DMB entry (RDE). This message might be issued for a DEDB with a large number of areas.

System action: Processing continues without storing the information about the DEDB areas after the area
that caused this condition. For these DEDB areas, the Integrity Checker processes nothing.

**User response:** None.

**FABL0203W** DMB MISMATCH FOUND FOR DBD: dddddddd [AREA: areaname]

**Explanation:** The Integrity Checker utility found a mismatch in the DMB information for the indicated resource while comparing the information in the RDE with the information in the ACB, DBD, or RECON. The resource (ddddd) is either a non-HALDB full-function database, HALDB partition, or a DEDB with area name (areaname).

Multiple FABL0203W messages are issued. Subsequent FABL0203W messages indicate the DMB information where the mismatch was found, the value in the RDE (RDE VALUE), and the value in the ACB or DBD (DBD VALUE or ACB VALUE). The DBD VALUE and the ACB VALUE might show a value that is obtained from the RECON data sets.

**System action:** Processing continues. The Integrity Checker utility causes the RDE for which the mismatch was found to expire, and creates a new RDE with the information in the ACB, DBD, or RECON.

**User response:** If the mismatch is unexpected, check whether the DBD, ACB, or RECON that you are using is correct.

**FABL0204E** DMB MISMATCH FOUND FOR DBD: dddddddd [AREA: areaname]

**Explanation:** The Integrity Checker utility found a mismatch in the DMB information for the indicated resource while comparing the information in the RDE with the information in the ACB, DBD, or RECON. The resource (ddddd) is either a non-HALDB full-function database, HALDB partition, or a DEDB with area name (areaname).

Multiple FABL0204E messages are issued. Subsequent FABL0204E messages indicate the DMB information where the mismatch was found, the value in the RDE (RDE VALUE), and the value in the ACB or DBD (DBD VALUE or ACB VALUE). The DBD VALUE and the ACB VALUE might show a value that is obtained from the RECON data sets.

**System action:** Processing continues. The Integrity Checker utility skips the database, partition, or area to obtain the database authorization. The Integrity Checker utility returns a nonzero return code for the requester of the database authorization to make the request for the database failure.

**User response:** If the mismatch is unexpected, check whether the DBD, ACB, or RECON that you are using is correct.

**FABL0205E** VERIFICATION PROCESS FOR dddddddd [RDE V] HAS BEEN STOPPED

**Explanation:** The Integrity Checker utility found a severe mismatch between RDE and either ACB or DBD and stopped the verification process for one of the following resources:

- Non-HALDB full-function database dddddddd
- HALDB partition dddddddd
- Area aaaaaaaa of database dddddddd

Multiple FABL0205E messages are issued. Other FABL0205E messages indicate the reason why the process was stopped.

**System action:** Processing continues without further verification for the database. If the verification option is 'W', the Integrity Checker utility causes the RDE for the mismatch found to expire, and creates a new RDE with the information in the DBD or ACB. If the verification option is 'D', the Integrity Checker utility skips the database, partition, or area to obtain the database authorization. The Integrity Checker utility returns a nonzero return code for the requester of the database authorization to make the request for the database failure.

**User response:** If the mismatch is unexpected, check whether the DBD or ACB you are using is correct.

**FABL0206E** NUMBER OF MESSAGES FOR dddddddd [RDE CREA] HAS REACHED UPPER THRESHOLD

**Explanation:** The Integrity Checker utility has detected that the number of messages issued for DMB mismatch for non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaa of database dddddddd has reached the upper threshold value. The Integrity Checker utility stops issuing further mismatch messages for the database.

**System action:** Processing continues.

**User response:** If you need to change the upper threshold number for the mismatch messages, generate a new global option module specifying a larger value for the VERIFYLMT= parameter of the FABLPGIN statement. If you need an unlimited number of mismatch messages, use 99.

**FABL0207E** RDE CREATION FAILED FOR DATABASE: dddddddd [AREA: aaaaaaaa]

**Explanation:** The Integrity Checker utility failed to create an RDE for non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaa of database dddddddd.

**System action:** Processing continues without creating the RDE.

**User response:** Check whether there are any errors for...
the LICON data set. If you find errors, correct the errors in the LICON data set and restart the IMS online subsystem or rerun the batch job.

**FABL0208W CHECKSUM MISMATCH FOUND FOR DBD: dddddddd [AREA: aaaaaaaa]**

**Explanation:** The Integrity Checker utility found a mismatch in the checksum value of the exit routines between that stored in RDE for non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaa of database dddddddd and that Integrity Checker calculated from the exit routines defined in ACB or DBD for this database, partition, or area. Multiple FABL0208W messages are issued. Other FABL0208W messages indicate the exit routine in which the mismatch was found and the module name for the exit routine.

**System action:** Processing continues. The Integrity Checker utility causes the RDE for which the mismatch was found to expire, and creates a new RDE with a checksum value for the exit routine.

**User response:** If the mismatch is unexpected, check whether the exit routine you are using is correct.

**FABL0209E CHECKSUM MISMATCH FOUND FOR DBD: dddddddd [AREA: aaaaaaaa]**

**Explanation:** The Integrity Checker utility found a mismatch in the checksum value of the exit routines between that stored in RDE for non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaa of database dddddddd and that the Integrity Checker utility calculated from the exit routines defined in ACB or DBD for this database, partition, or area. Multiple FABL0209E messages are issued. Other FABL0209E messages indicate the exit routine in which the mismatch was found and the module name for the exit routine.

**System action:** Processing continues. The Integrity Checker utility skips this database, partition, or area to obtain the database authorization. The Integrity Checker utility returns a nonzero return code for the requester of the database authorization to make the request for the database failure with the reason code $\$$. 

**User response:** If the mismatch is unexpected, check whether the exit routine you are using is correct.

**FABL0210E MODULE NOT FOUND FOR module**

**Explanation:** Module module was not found in the STEPLIB concatenation.

**System action:** The IMS online subsystem or the IMS batch job continues processing without computing the checksum value for the module. The LICON utility ends with a return code of 8.

**User response:** Check whether the module is provided for the STEPLIB concatenation of the job. Supply the module to the STEPLIB concatenation and rerun the job.

**FABL0211E FIND FAILED FOR DDNAME: ddname MODULE: module. RC=rr**

**Explanation:** The FIND macro failed for the indicated module in the data set that is indicated by ddname. The return code is rr.

**System action:** The IMS online subsystem or the IMS batch job continues processing without computing the checksum value for the module. The LICON utility ends with a return code of 8.

**User response:** Check whether the correct data set is specified on the DD statement. Correct the error and rerun the job.

**FABL0212W CHECKSUM CALCULATION WAS SKIPPED FOR MODULE: module IN DBD: dddddddd [AREA: aaaaaaaa]**

**Explanation:** The Integrity Checker utility cannot calculate the checksum value for module module that is specified in either ACB or DBD for non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaa of database dddddddd.

**System action:** Processing continues without storing the checksum value for the module into RDE.

**User response:** Check the preceding messages that explain the warning conditions. If this message is issued in a Fast Path Advanced Tool job of IMS HP Fast Path Utilities, see the topic "Considerations on using the Integrity Checker utility" in the IMS Fast Path Solution Pack: IMS HP Fast Path Utilities User’s Guide for the cause of this warning message.

**FABL0213W CHECKSUM VALUE NOT FOUND FOR MODULE: module IN RDE FOR DATABASE: dddddddd [AREA: aaaaaaaa]**

**Explanation:** The Integrity Checker utility found no checksum value for module module in the RDE for non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaa of database dddddddd, because the RDE has been created without the checksum value.

**System action:** Processing continues without verifying the checksum value for the module. In the IMS online environment or in the batch environment, the Integrity Checker utility expires the RDE in which no checksum value was stored and creates a new RDE with the checksum value.

**User response:** None.
FABL0214W - CHECKSUM VERIFICATION WAS SKIPPED FOR MODULE: module IN DBD: dddddddd [AREA: aaaaaaaaa]

Explanation: The Integrity Checker utility skips the verification process for the checksum value for module module that is specified in either ACB or DBD for non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaaa of database dddddddd.

System action: Processing continues without verifying the checksum value for the module.

User response: Check the preceding messages that explain the warning conditions.

FABL0215E - READ ERROR FOR DDNAME: ddname MODULE: module

Explanation: The READ macro failed for the indicated module in the data set that is indicated by ddname.

System action: The IMS online subsystem or the IMS batch job continues without computing the checksum value for the module. The LICON utility ends with a return code of 8.

User response: Check whether the correct data set is specified on the DD statement. Correct the error and rerun the job.

FABL0216W - NAME/TOKEN SERVICE service FAILED. NAME: nametoken RC=nn

Explanation: The process failed in the z/OS MVS Name/Token Service. service shows the service name. nn is the return code of the Name/Token service.

System action: Processing continues even if a database is reloaded from unload data sets in a compressed format.

User response: Find the cause of the error. For the return code, see z/OS MVS Programming: Assembler Services Reference, Volume 2 (JARR2V-XCTLX). If necessary, correct the warning condition and rerun the job.

FABL0217I - VERIFICATION OF COMPRESSION ROUTINE CHANGE IS NOT ENABLED

Explanation: Because IMS HP Load is not in the required maintenance level, the Integrity Checker utility could not verify the following changes of the segment edit/compression exit routines during the reorganization:

- Changes in the COMPRTN= parameters in the DBD definitions
- Changes in the logic of segment edit/compression exit routines

System action: Processing continues.

User response: If you want to validate the changes of the segment edit/compression exit routines, apply the corresponding PTF of APAR PK61325 to IMS High Performance Load and rerun the IMS High Performance Load job.

FABL0219I - RDE FORMAT LEVEL WILL BE UPGRADED AUTOMATICALLY FOR DBD: dddddddd [AREA: aaaaaaaaa]

Explanation: The format level of the RDE for the indicated resource is outdated. When verification of the RDE completes, the Integrity Checker utility upgrades the format level of the RDE by resetting the reserved areas, and then issues an FABL0201I message. However, if the verification is done by using the VERIFY.DB command of the LICON utility, the Integrity Checker utility will not upgrade the RDE.

Until the Integrity Checker utility upgrades the RDE and message FABL0201I is issued, the following functions are disabled:

- Verification of logic changes in the randomizer, the compression routine, and the partition selection exit
- Verification of the CRTE section of the indexed DEDB
- Restoration of RDEs during database recovery jobs

System action: Processing continues.

User response: None. This message is informational.

FABL0301E - LICON DATA SET IS EMPTY

Explanation: The LICON data set is empty. You need to initialize it before you use it.

System action: The IMS online subsystem or the IMS batch job ends abnormally. The LICON utility ends with a return code of 16.

User response: Check whether you are using the correct LICON data set. Initialize the LICON data set and restart the IMS online subsystem or rerun the batch job.

FABL0302E - LICON DATA SET IS NOT EMPTY

Explanation: You are attempting to initialize the LICON data set with the INIT.LICON command, but the LICON data set is not empty.

System action: The LICON utility ends with a return code of 16.

User response: Check whether you are using the correct LICON data set. Delete then define the LICON data set before initializing it. Rerun the LICON utility job requesting the INIT.LICON command.
FABL0303E  SNAP FAILED FOR DDNAME: ddbname

Explanation: The SNAP macro issued to data set whose DD name is ddbname failed.
System action: Processing continues without using the indicated data set.
User response: Check the status of the data set ddbname. Correct the error conditions, and restart the IMS online subsystem.

FABL0400I  LICON UTILITY COMMAND PROCESSING COMPLETE. HIGHEST RC = 00

Explanation: The LICON utility command processing has been completed successfully.
System action: The LICON utility ends normally.
User response: None. This message is informational.

FABL0401W  LICON UTILITY COMMAND PROCESSING ENDED WITH WARNINGS. HIGHEST RC = 04

Explanation: The LICON utility command processing has ended with warning messages.
System action: The LICON utility ends with a return code of 4.
User response: For details, check the warning message issued during command processing. Correct the warning condition. If necessary, rerun the LICON utility job.

FABL0402E  LICON UTILITY COMMAND PROCESSING ENDED WITH ERRORS. HIGHEST RC = 08

Explanation: The LICON utility command processing has ended with error messages.
System action: The LICON utility ends with a return code of 8.
User response: Find the error message that was issued during command processing. Correct the error, and rerun the LICON utility job.

FABL0403E  SEVERE ERROR. LICON UTILITY COMMAND PROCESSING ABORTED. RC = 16

Explanation: The LICON utility command processing has been canceled because a severe error occurred.
System action: The LICON utility ends with a return code of 16.
User response: Find the error message that was issued during command processing. Correct the error, and rerun the LICON utility job.

FABL0405E  INCORRECT EXEC PARM KEYWORD: text

Explanation: An incorrect keyword is specified for the EXEC parameter of the LICON utility job.
System action: The LICON utility ends immediately with a return code of 16. Processing is canceled.
User response: Check whether the EXEC parameter string is correct. Correct the error, and rerun the LICON utility job.

FABL0406E  DD NOT FOUND FOR dddddddd

Explanation: The required DD statement dddddddd was not found in the JCL of the LICON utility.
System action: The LICON utility ends immediately with a return code of 16. Processing is canceled.
User response: Check the JCL. Correct the error, and rerun the LICON utility job.

FABL0407E  DUMMY SPECIFIED FOR dddddddd

Explanation: DUMMY DD is specified for the required dddddddd DD statement.
System action: The LICON utility ends immediately with a return code of 16. Processing is canceled.
User response: Check the DD statement in the JCL. Correct the error, and rerun the LICON utility job.

FABL0408E  READ ERROR FOR TTRC X'ttttrrrc' RC=rr

Explanation: During the ACB member processing, a read error was detected for the ACB member whose TTRC is X'ttttrrrc'. rr is the return code of the READ macro.
System action: Processing continues by skipping the ACB member for which the read error was detected.
User response: Check whether the correct ACBLIB data set is used. Correct the error, and rerun the LICON utility job.

FABL0409E  UNSUPPORTED LEVEL OF IMS IS BEING USED: xx.x

Explanation: The utility is run under an unsupported version of IMS. xx.x is the version and release of IMS.
System action: The LICON utility ends immediately with a return code of 16.
User response: Check whether the version of IMS is correct. Correct the error, and rerun the LICON utility job.
### FABL0410I
**COMMAND COMPLETED WITH RC = 00**

**Explanation:** The LICON utility has processed the requested command successfully.

**System action:** Processing continues.

**User response:** None. This message is informational.

---

### FABL0411W
**COMMAND COMPLETED WITH RC = 04**

**Explanation:** The LICON utility has processed the requested command with warnings. Warning messages were issued for some of the databases for which processing was requested.

**System action:** Processing continues. A job step return code of 04 is set if a higher code has not been set.

**User response:** Find the associated warning messages. Correct the warning condition. If necessary, rerun the job.

---

### FABL0412E
**COMMAND COMPLETED WITH RC = 08**

**Explanation:** The LICON utility has processed the requested command with errors. Error messages were issued and processing failed for some of the databases for which processing was requested.

**System action:** Processing continues. A job step return code of 08 is set if a higher code has not been set.

**User response:** Find the associated error messages. Correct the error, and rerun the job.

---

### FABL0413E
**COMMAND COMPLETED WITH RC = 16**

**Explanation:** The LICON utility processing was canceled by a severe error. Error messages were issued and processing failed for some or all of the databases for which processing was requested.

**System action:** The LICON utility ends immediately with a job step return code of 16.

**User response:** Find the associated error messages. Correct the error, and rerun the job.

---

### FABL0414I
**THE FOLLOWING STATEMENTS SKIPPED:**

**Explanation:** This message shows the input statements that were skipped when the LICON utility processing was canceled by a severe error. The echo of the skipped control statements follows.

**System action:** The cancel processing continues. The job step return code 16 has already been set.

**User response:** None. This message is informational.

---

### FABL0420E
**INPUT STREAM END-OF-FILE FOUND BEFORE END OF COMMAND**

**Explanation:** An unexpected end-of-file was detected by the LICON utility for the input stream.

**System action:** The LICON utility ends with a job step return code of 8.

**User response:** Check whether correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

---

### FABL0421E
**REQUIRED OPTION NOT SPECIFIED**

**Explanation:** A required option keyword is not specified in the control statement.

**System action:** The LICON utility ends immediately with a job step return code of 16.

**User response:** Check whether the correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

---

### FABL0422E
**INPUT STREAM SYNTAX ERROR**

**Explanation:** A syntax error was detected in the input stream.

**System action:** The LICON utility ends immediately with a job step return code of 16.

**User response:** Check whether the correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

---

### FABL0423E
**INCORRECT COMMAND NAME SPECIFIED**

**Explanation:** An incorrect command name was detected in the input stream.

**System action:** The LICON utility ends immediately with a job step return code of 16.

**User response:** Check whether the correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

---

### FABL0424E
**INCORRECT OPTION SPECIFIED**

**Explanation:** An incorrect option was detected in the input stream.

**System action:** The LICON utility ends immediately with a job step return code of 16.

**User response:** Check whether the correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

---

Chapter 14. Troubleshooting 459
FABL0425E  •  FABL0442E

FABL0425E  DUPLICATE OPTION SPECIFIED
Explanation: Duplicate option specification was detected in the input stream.
System action: The LICON utility ends immediately with a job step return code of 16.
User response: Check whether the correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

FABL0426E  PARENTHESES MISSING
Explanation: The required parenthesis was not found for the input stream.
System action: The LICON utility ends immediately with a job step return code of 16.
User response: Check whether the correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

FABL0427E  VALUE value EXCEEDS LENGTH LIMIT
Explanation: The specified value value exceeds its length limit.
System action: The LICON utility ends immediately with a job step return code of 16.
User response: Check whether the correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

FABL0428E  VALUE value CONTAINS INCORRECT CHARACTER
Explanation: The specified value value contains an incorrect character.
System action: The LICON utility ends immediately with a job step return code of 16.
User response: Check whether the correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

FABL0429E  VALUE value INCORRECT
Explanation: The specified value value is incorrect.
System action: The LICON utility ends immediately with a job step return code of 16.
User response: Check whether the correct input control statements are supplied. Correct the error, and rerun the LICON utility job.

FABL0430E  INCORRECT TIMESTAMP SPECIFIED: tttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttt
have the current RDE for this database or area, and are
going to create one with the INIT.DB command, you
need to specify the REPLACE option.

**System action:** The utility skips processing this
database or area and tries to process the next member
if there is one. A job step return code of 8 is set.

**User response:** Check whether the REPLACE option is
correctly supplied. Specify the REPLACE option if you
had intended to do so. Rerun the LICON utility job.

---

**FABL0443E** ACB/DBD IN ERROR FOR DATABASE:
dddddddd

**Explanation:** The ACB or DBD member for database
dddddddd is in error. It cannot be processed.

**System action:** The utility skips processing database
dddddddd and tries to process the next member if there
is one. A job step return code of 8 is set.

**User response:** Check whether the correct ACB or
DBD member is used for the job. Correct the error, and
rerun the job.

---

**FABL0444E** DATABASE: dddddddd [AREA: aaaaaaaa] VERIFICATION FAILED

**Explanation:** The VERIFY.DB command detects data
mismatch between the RDE and the specified DBD (or
ACB). The detail of the mismatch is explained by the
messages that precede message FABL0204E for
non-HALDB full-function database or HALDB partition
dddddddd, or area aaaaaaaa of database dddddddd.

**System action:** The utility tries to process the next
member if there is one. A job step return code of 8 is
set.

**User response:** Check whether the correct ACB or
DBD member is used for the job. Correct the error, and
rerun the utility job.

---

**FABL0447E** RDE CREATION FAILED FOR DATABASE: dddddddd [AREA: aaaaaaaa]

**Explanation:** The INIT.DB command failed to create
the RDE for non-HALDB full-function database or
HALDB partition dddddddd or area aaaaaaaa of database
dddddddd.

**System action:** The utility skips processing this
database or area and tries to process the next member
if there is one. A job step return code of 8 is set.

**User response:** Find associated FABL messages that
show why the RDE creation failed. Correct the error,
and rerun the utility job.

---

**FABL0448E** NONE OF DATABASES SUCCESSFULLY PROCESSED

**Explanation:** At the end of processing a LICON utility
command, it turned out that none of the databases had
been successfully processed.

**System action:** The LICON utility ends immediately
with a job step return code of 16. The commands
following the current one are all skipped.

**User response:** Find associated FABL messages that
show the reason for the processing errors. Correct the
error, and rerun the utility job.

---

**FABL0449E** NEITHER ACBLIB NOR DBDLIB AVAILABLE

**Explanation:** Neither an ACBLIB DD statement nor a
DBDLIB DD one is supplied for the LICON utility JCL.
At least one of them is required for this command
processing.

**System action:** The LICON utility ends immediately
with a job step return code of 16. The commands
following the current one are all skipped.

**User response:** Check whether ACBLIB or DBDLIB
DD statement is correctly specified. Correct the error,
and rerun the utility job.

---

**FABL0450E** NO ACB/DBD MEMBER MATCHED FOR SPECIFIED DB NAME PATTERN: pattern

**Explanation:** No ACB or DBD member matched the
specified database name pattern pattern. Thus no ACB
or DBD member is processed for database pattern or
HALDB partition pattern.

**System action:** The LICON utility ends immediately
with a job step return code of 16. The commands
following the current one are all skipped.

**User response:** Check whether the database name
pattern supplied is correct. Correct the error, and rerun the utility job.

---

**FABL0451E** ACB/DBD MEMBER NOT FOUND FOR DATABASE: dddddddd

**Explanation:** The ACB or DBD member is not found
for the specified database or HALDB partition
dddddddd.

**System action:** The LICON utility ends immediately
with a job step return code of 16. The commands
following the current one are all skipped.

**User response:** Check whether the ACB or DBD
member exists in the specified ACBLIB or DBDLIB data
set. Correct the error, and rerun the utility job.
**FABL0452E**  
**ACBLIB/DBDLIB IS EMPTY**

**Explanation:** The ACBLIB or DBDLIB specified contains no member.

**System action:** The LICON utility ends immediately with a job step return code of 16. The commands following the current one are all skipped.

**User response:** Check whether the correct ACBLIB or DBDLIB DD statement is supplied. Correct the error, and rerun the utility job.

---

**FABL0453I**  
**DATABASE: dddddd NOT PROCESSED. REASON: text**

**Explanation:** Database dddddd was not processed. The reason is shown in text.

**System action:** The utility skips processing database dddddd and tries to process the next member if there is one.

**User response:** If INCOMPAT DMB is shown, ensure that the correct ACBLIB is used for the run.

---

**FABL0460E**  
**NO RDE FOUND FOR SPECIFIED [DB NAME PATTERN: pattern | DB NAME: dddddd AREA NAME PATTERN: pattern]**

**Explanation:** No RDE whose database name matches the specified database name pattern pattern or whose area name of database matches the specified area name pattern pattern was found.

**System action:** The LICON utility ends immediately with a job step return code of 16. The commands following the current one are all skipped.

**User response:** Check whether the correct database name pattern is supplied. Correct the error, and rerun the utility job.

---

**FABL0461E**  
**NO RDE FOUND FOR SPECIFIED [DB NAME PATTERN: pattern | DB NAME: dbname AREA NAME PATTERN: pattern] AND TIMESTMP:  

**Explanation:** No RDE that has time stamp  and whose database name matches the specified database name pattern pattern or whose area name of database matches the specified area name pattern pattern was found.

**System action:** The LICON utility ends immediately with a job step return code of 16. The commands following the current one are all skipped.

**User response:** Check whether the correct database name pattern is supplied. Correct the error, and rerun the utility job.

---

**FABL0462E**  
**NO RDE FOUND FOR DATABASE: 

**Explanation:** No RDE was found for non-HALDB full-function database or HALDB partition dddddd or area aaaaaaa of database dddddd.

**System action:** The LICON utility ends immediately with a job step return code of 16. The commands following the current one are all skipped.

**User response:** Check whether the correct database name is supplied. Correct the error, and rerun the utility job.

---

**FABL0463E**  
**NO RDE FOUND FOR DATABASE: 

**Explanation:** No RDE was found for non-HALDB full-function database or HALDB partition dddddd or area aaaaaaa of database dbname and time stamp.

**System action:** The LICON utility ends immediately with a job step return code of 16. The commands following the current one are all skipped.

**User response:** Check whether the correct database name is supplied. Correct the error, and rerun the utility job.

---

**FABL0470I**  
**START PROCESSING WITH ACBLIB/DBDLIB**

**Explanation:** This message shows that the LICON utility started processing with ACBLIB or DBDLIB.

**System action:** Processing continues.

**User response:** None. This message is informational.
FABL0480E  DBRC COMMAND FAILED

Explanation: An error occurred when the DBRC utility DSPURXRT was called.

System action: The LICON utility writes the messages of the DBRC utility in the FABLPRNT data set, and ends immediately with a job step return code of 16.

User response: Check the messages of the DBRC utility printed in the FABLPRNT data set. Correct the error, and rerun the job.

FABL0481E  NO DATABASE dddddddd FOUND IN RECON

Explanation: The LICON utility attempted to obtain names of the partitions associated with database dddddddd from the RECON data sets. However, the specified database was not found in RECON data sets.

System action: The LICON utility ends immediately with a job step return code of 16.

User response: Check whether the correct RECON data sets are used. Correct the error, and rerun the job. If the correct RECON data sets are used, specify the HALDB partition name as the DBD parameter, and rerun the job.

FABL0482E  NO PARTITIONS REGISTERED FOR THE DATABASE dddddddd IN RECON

Explanation: The LICON utility attempted to obtain names of the partitions associated with database dddddddd from the RECON data sets. However, there were no information about the partitions related to the specified database in the RECON data sets.

System action: The LICON utility ends immediately with a job step return code of 16.

User response: Check whether the correct RECON data sets are used. Correct the error, and rerun the job. If the correct RECON data sets are used, specify the HALDB partition name as the DBD parameter, and rerun the job.

FABL0483E  A MISMATCH TYPE FOUND BETWEEN DBD/ACB mmmmmmmmm AND RECON

Explanation: A mismatch of the database type was found between the DBD or the ACB mmmmmmmmm and the RECON record. The database type was defined as a HALDB in the DBD or the ACB. However, in the RECON data sets, it was defined as a non-HALDB.

System action: The LICON utility ends immediately with a job step return code of 16.

User response: Check whether the correct DBDLIB, ACBLIB, or RECON data sets are used. Correct the error, and rerun the job.

FABL0484E  NO AREA aaaaaaaa IS DEFINED FOR DATABASE dddddddd

Explanation: The LICON utility attempted to process area aaaaaaaa, which is associated with database dddddddd. However, the specified area was not found in the specified database.

System action: The LICON utility ends immediately with a job step return code of 16.

User response: Check whether the correct DBD or ACB is supplied in DBDLIB or ACBLIB. Correct the error, and rerun the LICON utility job.

FABL0485E  NO AREA NAME MATCHED THE SPECIFIED AREA NAME PATTERN; pattern IN DATABASE dddddddd

Explanation: No area name that matches the specified area name pattern pattern was found in the specified database dddddddd.

System action: The LICON utility ends immediately with a job step return code of 16.

User response: Check whether the correct DBD or ACB is supplied in DBDLIB or ACBLIB. Correct the error, and rerun the LICON utility job.

FABL0486E  THE SPECIFIED DATABASE dddddddd IS AN INCORRECT DATABASE ORGANIZATION

Explanation: The specified database dddddddd is not a DEDB although the AREA option is specified.

System action: The LICON utility ends immediately with a job step return code of 16.

User response: Check whether the DBD name is correct. Correct the error, and rerun the LICON utility job.

FABL0487E  THE INPUT LICON DATA SET IS INCORRECT

Explanation: The LICON data set has records of unsupported format. The records might be in the V1 format.

System action: The LICON utility ends immediately with a job step return code of 16.

User response: Check whether the LICON data set has V1 format records. If the data set has V1 format records, delete the LICON data set, re-create the LICON data set, and rerun the job.
FABL0600E • FABL0651E

FABL0600E  UNSUPPORTED LEVEL OF IMS IS BEING USED: xx.x

Explanation: The Integrity Checker utility is run under an unsupported version of IMS. xx.x is the version and release of IMS.

System action: The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.

User response: Check whether the version of IMS is correct. Correct the error, and rerun the batch job.

FABL0601I  NO RDE IS FOUND FOR DATABASE: dddddddd [AREA: aaaaaaaa] TIMESTAMP: yyyyddd hh:mm:ss.thmiju

Explanation: The Integrity Checker utility searched for the valid RDE for the DMB verification of non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaa of database dddddddd at UTC time stamp yyyyddd hh:mm:ss.thmiju, but could not find it.

System action: Processing continues. The Integrity Checker utility causes the current RDE to expire if the RDE exists, and creates a new RDE with the information in the DBD.

User response: None. This message is informational.

FABL0602I  RDE IS FOUND FOR DATABASE: dddddddd [AREA: aaaaaaaa] TIMESTAMP: yyyyddd hh:mm:ss.thmiju

Explanation: The Integrity Checker utility found the valid RDE for the DMB verification of non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaa of database dddddddd at UTC time stamp yyyyddd hh:mm:ss.thmiju.

System action: Processing continues. The Integrity Checker utility will use the RDE to do the DMB verification from now on.

User response: None. This message is informational.

FABL0603E  AN INPUT PARAMETER OF api IS NOT CORRECT (PARM: parameter/FUNC=xxxxxx)

Explanation: An incorrect input parameter is specified.

System action: The Integrity Checker utility returns a nonzero return code to the IMS Library Integrity Utilities service requester.

User response: None.

FABL0604E  RDE CREATION FAILED FOR DATABASE: dddddddd [AREA: aaaaaaaa]

Explanation: The Integrity Checker utility failed to create an RDE for non-HALDB full-function database or HALDB partition dddddddd or area aaaaaaaa of database dddddddd.

System action: The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.

User response: None.

FABL0605I  RDE IS NOT CREATED BECAUSE DMB MISMATCH IS FOUND FOR DBD: dbdname [AREA: areaname]

Explanation: The Integrity Checker utility was requested to create an RDE for the indicated database requested by the requester of the IMS Library Integrity Utilities service. However, the requester of the service does not change the DBD definition and specifies that an RDE is not created when a DMB mismatch is found. This message indicates that a DMB mismatch was found between the RDE and either ACB or DBD, and that RDE was not created. When the indicated DBD is for a DEDB, the name of the area is also shown in the message.

System action: Processing continues. The Integrity Checker utility returns return code 0 and reason code 4 to the requester of the IMS Library Integrity Utilities service.

User response: If the mismatch is unexpected, check whether the DBD or ACB that you are using is correct.

FABL0606E  NAME/TOKEN SERVICE service FAILED. NAME: nametoken RC=rc

Explanation: The process failed in the z/OS MVS Name/TOKEN Service. service shows the service name. rc shows the return code of the Name/TOKEN service.

System action: The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.

User response: Find the cause of the error. For the explanation of the return code, see z/OS MVS Programming: Assembler Services Reference, Volume 2 (IAR2V-XCTLX). If necessary, correct the error and rerun the job.

FABL0651E  GETMAIN FAILED FOR SIZE=size

Explanation: The GETMAIN macro for storage (size=size) failed.

System action:
• The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.
• The LICON utility ends immediately with a job step return code of 16.

User response: Determine if the REGION parameter for the JOB or the EXEC statement is large enough. Increase the region size and rerun the batch job or the utility job.

FABL0652E  **DEVTYPE FAILED FOR DDNAME:**

  `ddname (RC=rc)`

**Explanation:** After a DEVTYPE macro was issued to get information about the device that is associated with `ddname`, the return code indicated that the attempt to do so was unsuccessful.

**System action:**

- The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.
- The LICON utility ends immediately with a job step return code of 16.

**User response:** Determine if the `ddname` in the DD statement specifies the correct data set. Correct the error, and rerun the batch job or the utility job.

FABL0653E  **VSAM macro FAILED FOR DDNAME:**

  `ddname RC=rc RSN=reason_code`

**Explanation:** The VSAM macro `macro` failed for the data set whose DD name is `ddname`. The return code is `rc`, and the reason code is `reason_code`.

**System action:**

- The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.
- The LICON utility ends immediately with a job step return code of 16.

**User response:** Determine if the correct VSAM data set is being used. For the return code and the reason code, see z/OS DFSMS Macro Instructions for Data Sets. Correct the error, and rerun the batch job or the utility job.

FABL0654E  **RDJFCB FAILED FOR DDNAME:**

  `ddname (RC=rc)`

**Explanation:** The RDJFCB macro failed for the DD name `ddname`. The return code is `rc`.

**System action:**

- The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.
- The LICON utility ends immediately with a job step return code of 16.

**User response:** This error is likely an internal system error. Contact IBM Software Support.

FABL0655E  **INCORRECT IMS RELEASE LEVEL RECON DATA SET IS USED FOR DDNAME:**

**Explanation:** The data set that was used for the DD name `ddname` has an incorrect IMS release level of the RECON data set.

**System action:**

- The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.
- The LICON utility ends immediately with a job step return code of 16.

**User response:** Specify the correct IMS release level of the RECON data set, and rerun the batch job or the utility job.

FABL0656E  **NO RECON HEADER RECORD IS FOUND IN RECON DATA SET:**

**Explanation:** The data set that was used for the DD name `ddname` is not a RECON data set.

**System action:**

- The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.
- The LICON utility ends immediately with a job step return code of 16.

**User response:** Specify the correct RECON data set, and rerun the batch job or the utility job.

FABL0657E  **NO RECON HEADER EXTENSION RECORD IS FOUND IN RECON DATA SET:**

**Explanation:** The data set that was used for the DD name `ddname` is not a RECON data set or has an incorrect IMS release level of the RECON data set.

**System action:**

- The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.
- The LICON utility ends immediately with a job step return code of 16.

**User response:** Specify the correct IMS release level of the RECON data set, and rerun the batch job or the utility job.

FABL0658E  **TWO VALID RECON DATA SETS ARE NOT PROVIDED**

**Explanation:** The two IMS release levels of RECON data sets are not correct.

**System action:**
• The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.
• The LICON utility ends immediately with a job step return code of 16.

User response: Specify two correct IMS release levels of the RECON data sets, and rerun the batch job or the utility job.

FABL0659E MINVERS IS DEFINED INCORRECTLY IN RECON DATA SET: 

ddbname

Explanation: The RECON data set has incorrect MINVERS information.

System action:
• The Integrity Checker utility returns a nonzero return code to the requester of the IMS Library Integrity Utilities service.
• The LICON utility ends immediately with a job step return code of 16.

User response: Specify the correct MINVERS for the RECON data set when initializing RECON, and rerun the batch job or the utility job.

FABL1001E INCORRECT FUNCTION CODE (xx) IN PARAMETER

Explanation: An incorrect function code was specified in the parameter. xx is the hexadecimal function code.

System action: IMS Library Integrity Utilities Interface returns return code 16 to the caller.

User response: This error is probably an internal system error. Collect the dump, and contact IBM Software Support.

FABL1024E ERROR LOADING MODULE module:
ABEND CODE=mmmm REASON CODE=mmmm

Explanation: An error occurred while load module module was being loaded. mmm is the hexadecimal system abend code, and mmmmm is the hexadecimal reason code.

System action: IMS Library Integrity Utilities Interface returns return code 16 to the caller.

User response: Determine the cause of the load error. Correct the problem, and rerun the program.

FABL1041E NO DATA IN ddbname1 [OR ddbname2]

Explanation: A utility of IMS Library Integrity Utilities detected some errors in the DBD or PSB member. The utility does not generate any data in SYSPRINT or SYSPUNCH data set. ddbname1 is the DD name of the SYSPRINT data set, and ddbname2 is the DD name of the SYSPUNCH data set.

System action: IMS Library Integrity Utilities continues processing. If one or more DBD or PSB members are decoded, the utility sets the return code to 4. If no DBD or PSB members are decoded, the utility sets the return code to 8.

User response: Check the messages issued by the utility.

FABL1061E ERROR OCCURRED IN module [USER | SYSTEM] COMPLETION CODE=uuuu( sscanf)

Explanation: An error occurred in module module. uuuuu is the user completion code, and sscanf is the hexadecimal system completion code.

System action: IMS Library Integrity Utilities Interface returns return code 16 to the caller.

User response: Determine the cause of the error. Correct the problem, and rerun the program.
FABL2001I  [DBD | PSB] TO BE PROCESSED IS member

Explanation: The name of the DBD or PSB member specified for processing.

System action: The Consistency Checker utility continues processing.

User response: None. This message is informational.

FABL2002I  CONSISTENCY CHECK WAS SUCCESSFUL FOR member

Explanation: The Check function ended successfully, and no inconsistency between the DBD or the PSB definition and the other IMS definitions for member was found.

System action: The Consistency Checker utility continues processing.

User response: None. This message is informational.

FABL2003E  CONSISTENCY CHECK FAILED FOR member

Explanation: The Check function ended successfully and inconsistency between the DBD or the PSB definition and the other IMS definitions for member was found.

System action: The Consistency Checker utility creates a check report and continues processing.

User response: None.

FABL2004I  CONSISTENCY CHECK WAS SKIPPED FOR ddname

Explanation: The Check function is not done for ddname, because its database organization is specified in the NOCHKORG parameter.

System action: The Consistency Checker utility does not create a check report and continues processing.

User response: None. This message is informational.

FABL2005E  THE NUMBER OF INCONSISTENT PCBs IN member HAS EXCEEDED THE THRESHOLD

Explanation: The number of the inconsistent PCBs in the PSB member has exceeded the threshold value specified in the PCBERRLMT parameter. The Consistency Checker utility stops checking and printing further PCBs in the PSB.

System action: The Consistency Checker utility continues processing.

User response: None.

FABL2006I  ddname DATA SET IS [SPECIFIED | NOT SPECIFIED]

Explanation: The ddname data set is either specified or not specified. ddname is the DD name of the ACBLIB, DFSMDA, MODBLKS, SYSRDDS, or the NSYSRDDS data set.

System action: The Consistency Checker utility continues processing.

User response: None. This message is informational.

FABL2007I  PARAMETER USED IS: keyword=value

Explanation: The Consistency Checker utility proceeds with keyword=value.

System action: The Consistency Checker utility continues processing.

User response: None. This message is informational.

FABL2009I  ddname DATA SET IS NOT USED BECAUSE DRD= parameter IS SPECIFIED

Explanation: The indicated data set was not used because DRD= parameter is specified.

System action: Processing continues.

User response: None. This message is informational.

FABL2010I  CONTROL CARD SUPPLIED IS: echo of control statement

Explanation: This message is the echo of the SYSIN control statement that is verified by this program.

System action: The Consistency Checker utility continues processing.

User response: None. This message is informational.

FABL2011E  INCORRECT STATEMENT IN SYSIN DATASET

Explanation: A control statement with an incorrect format was found in the SYSIN data set.

System action: The Consistency Checker utility ends with a return code of 12.

User response: Correct the format of the control statement, and rerun the job.

FABL2012E  INCORRECT MEMBER NAME IS SPECIFIED

Explanation: The member name specified in the control statement was incorrect. For example, the member name contained more than eight characters.

System action: The Consistency Checker utility ends with a return code of 12.
**FABL2013E** Incorrect member name in the control statement, and rerun the job.

**FABL2014E** Duplicate statement is specified

**Explanation:** Two or more identical control statements were found in the SYSIN data set.

**System action:** The Consistency Checker utility ends with a return code of 12.

**User response:** Remove the duplicate statement, and rerun the job.

**FABL2015E** No data in SYSIN

**Explanation:** No control statement is found in the SYSIN data set.

**System action:** The Consistency Checker utility ends with a return code of 12.

**User response:** Specify the control statements in the SYSIN data set, and rerun the job.

**FABL2016E** Maximum SYSIN cards exceeded

**Explanation:** The number of control statements has exceeded the maximum value of 9999.

**System action:** The Consistency Checker utility ends with a return code of 12.

**User response:** Rerun the ignored cards.

**FABL2017E** Unsupported control statement is specified under IMS version x.x

**Explanation:** The indicated control statement is not supported under IMS Version x.x.

**System action:** The Consistency Checker utility ends with a return code of 12.

**User response:** Check the IMS version of DFSRESLB.

**FABL2019E** ddname data set is not used because it does not contain all the resource definitions

**Explanation:** The specified data set cannot be used because it does not contain all the resource definitions.

**System action:** Consistency Checker ends abnormally.

**User response:** Ensure that the data sets specified on the indicated DD statement are correct. If necessary, correct the data sets and rerun the job.

**FABL2020E** No member found for member in [DBDLIB | PSBLIB]

**Explanation:** The specified member member, or all of the members specified by a wildcard, were not found in the DBD or the PSB library.

**System action:** The Consistency Checker utility continues processing without reporting for this DBD or PSB member.

**User response:** Check whether member is correct. Correct the problem, and rerun the job.

**FABL2021E** Member was not [DBD | PSB] type ACB member

**Explanation:** The specified member was not a DBD-type or a PSB-type ACB member.

**System action:** The Consistency Checker utility creates a check report and continues processing.

**User response:** None.

**FABL2022E** IMS version inconsistent between ACB x.x and DFSRESLB y.y

**Explanation:** ACB x.x and DFSRESLB y.y use different versions of IMS.

**System action:** The Consistency Checker utility creates a check report and continues processing.

**User response:** Check the IMS version used for ACB and DFSRESLB.
FABL2024E DUPLICATE MEMBER FOUND FOR

\textit{member IN [DBDLIB | PSBLIB] \(+nnn\)}

\textbf{Explanation:} Duplicate member was found in DBDLIB or PSBLIB. The \textit{member} is the specified member name. \(+nnn\) is the relative position of a data set within a concatenation of data sets.

\textbf{System action:} The Consistency Checker utility continues processing without checking this DBD or PSB member.

\textbf{User response:} Check the duplicate DBD or PSB member.

FABL2025E RDJFCB MACRO FAILED FOR \textit{ddname: RETURN CODE=rrrr}

\textbf{Explanation:} The RDJFCB macro attempted for the indicated DD, but failed.

\textbf{System action:} Consistency Checker ends abnormally.

\textbf{User response:} Check the cause of the error. Correct the problem, and rerun the program.

FABL2026I THE DATA SET THAT IS SPECIFIED BY NSYSRDDS DD IS A SYSTEM RDDS

\textbf{Explanation:} The RDDS specified on the NSYSRDDS DD is a system RDDS.

\textbf{System action:} Consistency Checker continues processing.

\textbf{User response:} None. This message is informational.

FABL2030E \textit{ddname} DID NOT OPEN

\textbf{Explanation:} The data set \textit{ddname} could not be opened during initialization.

\textbf{System action:} The Consistency Checker utility ends abnormally.

\textbf{User response:} Check the cause of this failure.

FABL2031E UNSUPPORTED IMS VERSION: \textit{x.x} for PSB CHECK PROCESS

\textbf{Explanation:} DFSRESLB IMS version \textit{x.x} is not supported for the consistency check of PSBs.

\textbf{System action:} The Consistency Checker utility ends with a return code of 12.

\textbf{User response:} Check the IMS version of DFSRESLB.

FABL2035E GETMAIN FAILED

\textbf{Explanation:} The program could not obtain enough area with the GETMAIN macro.

\textbf{System action:} The Consistency Checker utility ends abnormally.

\textbf{User response:} If the specified region size is too small, increase the REGION size in the JOB statement in the JCL, and rerun the program.

FABL2040E ERROR LOADING [DBDLIB | PSBLIB] \textit{member: (ABEND CODE=nnnn REASON CODE=mmmm)}

\textbf{Explanation:} An error has occurred during DBD or PSB \textit{member} load. \textit{nnnn} is the hexadecimal system abend code, and \textit{mmmm} is the hexadecimal reason code.

\textbf{System action:} The Consistency Checker utility continues processing without reporting for this DBD or PSB member.

\textbf{User response:} Check the cause of this load error. Correct the problem, and rerun the program.
FABL2041E  ERROR LOADING dname member:  
(ABEND CODE=mnmm  REASON  
CODE=mmmm)

Explanation: An error occurred during dname member  
load. mnnn is the hexadecimal system abend code, and  
mmmm is the hexadecimal reason code.

System action: The Consistency Checker utility creates  
a check report and continues processing.

User response: Check the cause of load error. Correct  
the problem, and rerun the program.

FABL2042E  ERROR LOADING dname member:  
(ABEND CODE=mnmm  REASON  
CODE=mmmm)

Explanation: An error occurred during dname member  
load. mnnn is the hexadecimal system abend code, and  
mmmm is the hexadecimal reason code.

System action: The Consistency Checker utility ends  
abnormally.

User response: Check the cause of the load error.  
Correct the problem, and rerun the program.

FABL2045E  READ ERROR ON [DBD | PSB]  
DIRECTORY

Explanation: A read error occurred while the DBD or  
the PSB directory was being read.

System action: The Consistency Checker utility ends  
abnormally.

User response: Check the cause of the read error.  
Correct the problem, and rerun the program.

FABL2046E  BLDL FAILED FOR ACB=member

Explanation: An error occurred while a BLDL macro  
for an ACB member was being issued.

System action: The Consistency Checker utility creates  
a check report and continues processing.

User response: Check the cause of the BLDL macro  
failure, correct it, and rerun the program.

FABL2047E  ERROR READING ACB=member

Explanation: An error occurred while an ACB member  
was being read.

System action: The Consistency Checker utility creates  
a check report and continues processing.

User response: Check the cause of the failure, correct  
it, and rerun the program.

FABL2050E  DYNAMIC [ALLOCATION |  
DEALLOCATION | CONCATENATE]  
FAILED FOR dname: RETURN  
CODE=xxxx, REASON CODE=yyyy

Explanation: An attempt to dynamically allocate,  
deallocate, or concatenate the dname data set failed.  
xxxx is the hexadecimal return code, and yyyy is the  
hexadecimal reason code.

System action: The Consistency Checker utility ends  
abnormally.

User response: This error is probably an internal  
system error. Collect the dump, and contact IBM  
Software Support.

FABL2051E  dname DID NOT OPEN

Explanation: The dname data set that was allocated  
dynamically could not be opened during initialization.

System action: The Consistency Checker utility ends  
abnormally.

User response: This error is probably an internal  
system error. Collect the dump, and contact IBM  
Software Support.

FABL2052E  DBRC COMMAND FAILED: RETURN  
CODE=mmmm

Explanation: The Consistency Checker utility  
internally linked DBRC utility DSPURXR to obtain  
DBRC information. mnnn is the hexadecimal return  
code. See the IMS Messages and Codes, Volume 4: IMS  
Component Codes.

System action: The Consistency Checker utility writes  
the messages of the DBRC utility in the SYSOUT data  
set, and ends abnormally.

User response: Check the cause of the error. Correct  
the problem, and rerun the program.

FABL2053W  UNSUPPORTED DBD member IN  
DBDLIB

Explanation: The specified DBD member in the DBD  
library is not supported for the Consistency Checker  
utility.

System action: The Consistency Checker utility skips  
this member and continues processing.

User response: Check the DBD member.

FABL2054E  EITHER dname1 DD OR dname2 DD  
MUST BE SPECIFIED

Explanation: DRD=YES is specified, but SYSRDDS DD  
or NSYSRDDS DD is not specified.

System action: Consistency Checker ends abnormally.
User response: Specify either SYSRDDS DD or NSYSRDDS DD, and rerun the program.

**FABL2055E**  A READ ERROR OCCURRED WHILE READING THE RDDS THAT WAS SPECIFIED BY dname DD

Explanation: An error occurred when the RDDS data set was read.

System action: Consistency Checker ends abnormally.

User response: Correct the problem, and rerun the program.

**FABL2056E**  THE DATA SET THAT IS SPECIFIED BY dname DD IS NOT A VALID RDDS

Explanation: The data set that is specified on the indicated DD statement is not in a valid RDDS format.

System action: Consistency Checker ends abnormally.

User response: Ensure that the data set specified on the DD statement is correct, and rerun the job.

**FABL2057E**  THE DATA SET THAT IS SPECIFIED BY SYSRDDS DD IS NOT A SYSTEM RDDS

Explanation: The RDDS that is specified on the SYSRDDS DD is not a system RDDS.

System action: Consistency Checker ends abnormally.

User response: Ensure that the data sets that are specified on the DD statement are correct, and rerun the job.

**FABL3000E**  DYNALLOC FAILED FOR DDNAME: dname DSNAME: dsn RC=rr RSN=ssss

Explanation: A dynamic allocation request for DDNAME dname DSNAME dsn failed. The return code is rr, and the reason code is ssss.

System action: The IMS online subsystem that has a BPE-based DBRC region continues processing with Integrity Checker deactivated. The IMS online subsystem that has a non-BPE-based DBRC region or the IMS batch job ends abnormally. The LICON utility ends immediately with a job step return code of 16.

User response: Find the reason for the dynamic allocation request failure. For the return code and the reason code, see the z/OS MVS Programming: Authorized Assembler Services Guide. Correct the error, and restart the IMS online subsystem or rerun the batch job. For the IMS online subsystem that has a BPE-based DBRC region, do not issue the BPE USEREXIT command until you restart the subsystem.

**FABL3001E**  LOAD FAILED FOR MODULE: module CODE=cccc RSN=ssss

Explanation: Load failed for module module. The return code is cccc, and the reason code is ssss.

System action: The IMS online subsystem that has a BPE-based DBRC region continues processing with Integrity Checker deactivated. The IMS online subsystem that has a non-BPE-based DBRC region or the IMS batch job ends abnormally. The LICON utility ends immediately with a job step return code of 16.

User response: Find the reason for the load failure. For the return code and the reason code, see z/OS MVS Programming: Assembler Services Reference, Volume 2 (IARRV2-XCTLX). Check whether the correct program libraries are concatenated to the STEPLIB DD statement. Correct the error, and restart the IMS online subsystem or rerun the batch job. For the IMS online subsystem that has a BPE-based DBRC region, do not issue the BPE USEREXIT command until you restart the subsystem.

**FABL3002E**  NO LICON DATA SET NAME GIVEN

Explanation: No LICON data set name is given by the IMS Library Integrity Utilities global option modules. You need to specify it at either the subsystem level (LIU@imsid or LIUG@imsid) or the installation level (LIU@INST or LIUG@INST) of the IMS Library Integrity Utilities global option modules.

System action: The IMS online subsystem that has a BPE-based DBRC region continues processing with Integrity Checker deactivated. The IMS online subsystem that has a non-BPE-based DBRC region or the IMS batch job ends abnormally. The LICON utility ends immediately with a job step return code of 16.

User response: Check whether the correct global option modules are used for the run. Check whether the data set name of the LICON data set is correctly supplied when the global option modules are defined. Correct the error, and restart IMS online subsystem or rerun the batch job. For the IMS online subsystem that has a BPE-based DBRC region, do not issue the BPE USEREXIT command until you restart the subsystem.

**FABL3003E**  INCORRECT LICON DATA SET NAME: dsn

Explanation: The data set name of the LICON data set is incorrect.

System action: The IMS online subsystem that has a BPE-based DBRC region continues processing with Integrity Checker deactivated. The IMS online subsystem that has a non-BPE-based DBRC region or the IMS batch job ends abnormally. The LICON utility ends immediately with a job step return code of 16.

User response: Check whether the data set name of the LICON data set was correctly supplied when the
global option modules are defined. Correct the error, and restart the IMS online subsystem or rerun the batch job. For the IMS online subsystem that has a BPE-based DBRC region, do not issue the BPE USEREXIT command until you restart the subsystem.

FABL3004E  NO GLOBAL OPTION MODULES FOUND

Explanation: No global option modules are found in the STEPLIB DD concatenation.

System action: The LICON utility ends immediately with a job step return code of 16.

User response: The library that contains the global option module must be concatenated to the STEPLIB DD. Check whether the program libraries are correctly concatenated to the STEPLIB DD. Correct the error, and rerun the LICON utility job.

FABL3005E  GLOBAL OPTION MODULE NOT FOUND: LIU@imsid

Explanation: The indicated global option module (or the global option module named LIUG@imsid) is not found in the STEPLIB DD concatenation.

System action: The LICON utility ends immediately with a job step return code of 16.

User response: The library that contains the global option module must be concatenated to the STEPLIB DD. Determine if the program libraries are correctly concatenated to the STEPLIB DD. Correct the error, and rerun the LICON utility job.

FABL3006E  VSAM OPEN FAILED FOR DDNAME: ddname

Explanation: The OPEN macro for the VSAM data set ddname failed. The return code is rr, and the reason code is sssss.

System action: The IMS online subsystem or the IMS batch job ends abnormally.

User response: Check whether the correct data set is used for the DD name. For the return code and the reason code of the OPEN macro, see z/OS DFSMS Macro Instructions for Data Sets. Correct the error, and restart the IMS online subsystem or rerun the batch job.

FABL3007E  DATA SET OPEN FAILED FOR DDNAME: ddname

Explanation: The OPEN macro for the data set ddname failed. The return code was rr.

System action: The IMS online subsystem that has a BPE-based DBRC region continues processing with Integrity Checker deactivated. The IMS online subsystem that has a non-BPE-based DBRC region or the IMS batch job ends abnormally.

User response: Verify that the exit list member that is specified on the EXITMBR statement is incorrect for Integrity Checker. The Integrity Checker module is not specified as a DBRC user exit routine.

System action: The IMS online subsystem continues processing with Integrity Checker deactivated.

User response: Correct the BPE configuration parameter member or the user exit list member that is specified by the EXITMBR statement. Correct the BPE configuration parameter member or the user exit list member, and restart the IMS online subsystem. For instructions for configuring IMS PROCLIB, see "Configuring for a BPE-based DBRC environment" on page 65. Do not issue the BPE
FABL3011E  THE DBRC USER EXIT LIST IS INCORRECT FOR LIU INTEGRITY CHECKER

Explanation: The DBRC user exit list member of the IMS PROCLIB data set is incorrect for the Integrity Checker utility. The Integrity Checker utility is not specified as a DBRC request user exit.

System action: The IMS online subsystem continues processing with the Integrity Checker utility deactivated.

User response: Verify that the FABLBIN0 module is specified by using the EXITDEF statement of the DBRC user exit list member in the IMS PROCLIB data set. Also verify that the module is specified as a DBRC request user exit. Correct the DBRC user exit list member and restart the IMS online subsystem. For instructions for configuring for IMS PROCLIB, see "Configuring for a BPE-based DBRC environment" on page 65. Do not issue the BPE USEREXIT command until you restart the subsystem.

FABL3012E  ESTAEX FAILED RC=rc RSN=rsn ID: id

Explanation: The ESTAEX macro failed. The return code is rc, and the reason code is rsn. id is an identifier that is associated with the internal location where the ESTAEX macro is issued.

System action: The IMS online subsystem continues processing with Integrity Checker deactivated.

User response: For the meaning of the return code and the reason code, see z/OS MVS Programming: Authorized Assembler Services Reference, Volume 2 (EDTINFO-IXGWRITE). Collect the dump and contact IBM Software Support. Do not issue the BPE USEREXIT command.

FABL3013E  UNSUPPORTED LEVEL OF IMS IS BEING USED: mn.n

Explanation: The Integrity Checker utility was run on an unsupported version of IMS. mn.n is the version and release of IMS.

System action: The IMS online subsystem continues processing with the Integrity Checker utility deactivated.

User response: Correct the error conditions and restart the IMS online subsystem. Do not issue the BPE USEREXIT command until you restart the subsystem.

FABL3014E  NAME/TOKEN SERVICE service FAILED. NAME: nametoken RC=rc

Explanation: The process failed in the z/OS MVS Name/Token Service. service shows the service name. rc is the return code of the Name/Token service.

System action: The IMS online subsystem continues processing with the Integrity Checker utility deactivated.

User response: Identify the cause of the error. For the return code, see z/OS MVS Programming: Assembler Services Reference, Volume 2 (IARR2V-XCTLX). Collect the dump, and contact IBM Software Support. Do not issue the BPE USEREXIT command.

FABL3015E  ENQ FOR NAME/TOKEN SERVICE AND ALL RETRIES FAILED. RC=rr

Explanation: The ENQ macro for the Name/Token service failed and the retry limit has been reached. rr is the ENQ macro return code from the last attempt.

System action: The IMS batch job ends abnormally.

User response: See the z/OS MVS Programming: Assembler Services Reference, Volume 1 (ABEND-HSPSERV) for the return code and identify the cause of the ENQ failure. Correct the error, and rerun the batch job.

FABL3016E  LOAD FAILED FOR MODULE DFSVC000 AND DFSBSCD0. RC=cccc RSN=ssss

Explanation: The LOAD macro failed to load the DFSVC000 module and the macro returned a return code of 0A06. Then the macro attempted to load the DFSBSCD0 module, but it failed with the return code and the reason code that are indicated in the message.

System action: The IMS batch job ends abnormally.

User response: See the z/OS MVS Programming: Assembler Services Reference, Volume 2 (IARR2V-XCTLX) for the return code and identify the cause of the LOAD failure. Also, ensure that the correct program libraries are concatenated to the STEPLIB DD statement. Correct the error, and rerun the batch job.

FABL3106E  LOAD FAILED FOR MODULE: module CODE=cccc RSN=ssss

Explanation: Load of the indicated module has failed. The return code is cccc, and the reason code is ssss.

System action: The IMS online subsystem that has a BPE-based DBRC region continues processing with Integrity Checker deactivated. The IMS online subsystem that has a non-BPE-based DBRC region or the IMS batch job ends abnormally.

User response: Find the reason for the load failure. For the return code and the reason code, see z/OS MVS Programming: Assembler Services Reference, Volume 2 (IARR2V-XCTLX). Check whether the correct program libraries are concatenated to the STEPLIB DD statement. Correct the error, and restart the IMS online
subsystem or rerun the batch job. For the IMS online subsystem that has a BPE-based DBRC region, do not issue the BPE USEREXIT command until you restart the subsystem.

**FABL3301E**  
**VSAM macro FAILED FOR DDNAME: ddname RC=rr RSN=ssss**

**Explanation:** The VSAM macro failed for the data set whose DD name is *ddname*. The return code is *rr*, and the reason code is *ssss*. FABL3301E is issued for the failure of VSAM macros requested for ACBs.

**System action:** The IMS online subsystem continues processing with the Integrity Checker function deactivated. The IMS batch job or the LICON utility job ends abnormally.

**User response:** Check whether the correct VSAM data set is used. Ensure that enough space is allocated for the VSAM data set. For the return code and the reason code, see z/OS DFSMS Macro Instructions for Data Sets. Correct the error, and rerun the batch job. If necessary, restart the IMS subsystem.

---

**FABL3302E**  
**VSAM macro FAILED FOR DDNAME: ddname RC=rr RSN=ssss**

**Explanation:** The VSAM macro failed for the data set whose DD name is *ddname*. The return code is *rr*, and the reason code is *ssss*. FABL3302E is issued for the failure of VSAM macros requested for RPLs.

**System action:** The IMS online subsystem continues processing with the Integrity Checker function deactivated. The IMS batch job or the LICON utility job ends abnormally.

**User response:** Check whether the correct VSAM data set is used. Ensure that enough space is allocated for the VSAM data set. For the return code and the reason code, see z/OS DFSMS Macro Instructions for Data Sets. Correct the error, and rerun the batch job. If necessary, restart the IMS subsystem.

---

**FABL3303E**  
**VSAM SHOWCB FAILED FOR DDNAME: ddname RC=rr RSN=ssss**

**Explanation:** The VSAM SHOWCB macro failed for the data set whose DD name is *ddname*. The return code is *rr*, and the reason code is *ssss*.

**System action:** The IMS online subsystem continues processing with the Integrity Checker function deactivated. The IMS batch job or the LICON utility job ends abnormally.

**User response:** Check whether the correct VSAM data set is used. For the return code and the reason code, see z/OS DFSMS Macro Instructions for Data Sets. Correct the error, and rerun the batch job. If necessary, restart the IMS subsystem.
<table>
<thead>
<tr>
<th>Module/Code</th>
<th>Explanation</th>
<th>System action</th>
<th>User response</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FABL3404E</td>
<td>Module not found for <em>module</em></td>
<td>The LICON utility job ends abnormally.</td>
<td>Check whether the correct program libraries are concatenated to the STEPLIB DD statement. Correct the error, and rerun the job.</td>
<td></td>
</tr>
<tr>
<td>FABL3405E</td>
<td>LOAD failed for module: <em>module</em></td>
<td>The LICON utility job ends abnormally.</td>
<td>Check whether the correct load module is contained in the program libraries concatenated to the STEPLIB DD statement. Correct the error, and rerun the job.</td>
<td></td>
</tr>
<tr>
<td>FABL3406E</td>
<td>GETMAIN failed for size=*, <em>RC=rr</em></td>
<td>The LICON utility job ends abnormally.</td>
<td>Ensure that the REGION parameter for the JOB or EXEC statement is reasonably large enough. If the region size is small, increase the size, and rerun the job.</td>
<td></td>
</tr>
<tr>
<td>FABL3407E</td>
<td>READ error for PDS directory of [ACBLIB</td>
<td>DBDLIB]</td>
<td>The LICON utility job ends abnormally.</td>
<td>Check whether the correct data set is specified to the ACBLIB DD or the DBDLIB DD. Correct the error, and rerun the LICON utility job.</td>
</tr>
<tr>
<td>FABL3408E</td>
<td>FIND failed for ACBLIB. <em>RC=rr</em></td>
<td>The LICON utility job ends abnormally.</td>
<td>The PDS directory of the ACBLIB might be corrupted. Check whether the correct data set is specified to the ACBLIB DD. Correct the error, and rerun the LICON utility job.</td>
<td></td>
</tr>
<tr>
<td>FABL3409E</td>
<td>VSAM SHOWCB failed for <em>DDNAME: ddbname</em> <em>RC=rr</em> <em>RSN=ssss</em></td>
<td>The LICON utility job ends abnormally.</td>
<td>Check whether the correct VSAM data set is used. For the return code and the reason code, see z/OS DFSMS Macro Instructions for Data Sets. Correct the error, and rerun the batch job. If necessary, restart the IMS subsystem.</td>
<td></td>
</tr>
<tr>
<td>FABL3410E</td>
<td>VSAM MODCB failed for <em>DDNAME: ddbname</em> <em>RC=rr</em> <em>RSN=ssss</em></td>
<td>The LICON utility job ends abnormally.</td>
<td>Check whether the correct VSAM data set is used. For the return code and the reason code, see z/OS DFSMS Macro Instructions for Data Sets. Correct the error, and rerun the LICON utility job.</td>
<td></td>
</tr>
<tr>
<td>FABL3411E</td>
<td>GET failed for <em>DDNAME: ddbname</em></td>
<td>The LICON utility job ends abnormally.</td>
<td>Check whether the correct data set is specified. Correct the error, and rerun the LICON utility job.</td>
<td></td>
</tr>
<tr>
<td>FABL3412E</td>
<td>PUT failed for <em>DDNAME: ddbname</em></td>
<td>The LICON utility job ends abnormally.</td>
<td>Check whether the correct data set is specified. Correct the error, and rerun the LICON utility job.</td>
<td></td>
</tr>
<tr>
<td>FABL3413E</td>
<td>BLDL for <em>ACB MEMBER</em> member failed</td>
<td>The LICON utility job ends abnormally.</td>
<td>Check whether the BLDL macro for the indicated ACB member has failed.</td>
<td></td>
</tr>
</tbody>
</table>
User response: Check whether the correct data set is specified. The PDS directory of the ACBLIB might be corrupted. Correct the error, and rerun the LICON utility job.

FABL3414E DYNALLOC FAILED FOR DDNAME: 
ddname RC=rr RSN=ssss
Explanation: A dynamic allocation request for the indicated DD name has failed. The return code is rr, and the reason code is ssss.
System action: The LICON utility ends abnormally.
User response: Find the reason for the dynamic allocation request failure. For the return code and the reason code, see the z/OS MVS Programming: Authorized Assembler Services Guide. Correct the error, and rerun the job.

FABL3415E LOAD FAILED FOR DBD MEMBER: 
member
Explanation: The LOAD macro for the DBD member member failed.
System action: The LICON utility job ends abnormally.
User response: Check whether the correct DBD member is contained in the DBDLIB DD statement. Correct the error, and rerun the job.

FABL3800E THE INPUT LICON DATA SET IS INCORRECT
Explanation: The input LICON data set is of an incorrect version.
System action: The LICON data set migration utility ends abnormally.
User response: Check whether the input LICON data set is of a correct version. Correct the error, and rerun the job.

FABL3900E RECON ACCESS FAILED. text
Explanation: An error was detected in the RECON access processing.
text provides additional information about the error:
• FUNC=function RETURN CODE=return_code
  REASON CODE=reason_code KEYS: DBD=dbname DDN=ddname KEYTYPE=keytype accessype
• DBRC LIST COMMAND IS NOT COMPLETED.
  RC=return_code
• SYSPRINT DD FOR DBRC LIST COMMAND IS SPECIFIED AS DUMMY
• INTERNAL ERROR OCCURRED
System action: The IMS batch job ends abnormally.

User response: Correct the error, and rerun the batch job.

FABL3901E NAME/TOKEN SERVICE service FAILED: nametoken RC=rc
Explanation: The process failed in the z/OS MVS Name/Token service. service shows the service name. rc shows the return code of the Name/Token service.
System action: The batch job ends abnormally.
User response: See the z/OS MVS Programming: Assembler Services Reference, Volume 2 (IARR2V-XCTLX) for the return code and identify the cause of the failure. If necessary, correct the error and rerun the job.

FABL3902E GETMAIN FAILED FOR SIZE=size
  RC=rc
Explanation: The GETMAIN macro for storage (size=size) failed. The return code is rc.
System action: The batch job ends abnormally.
User response: See the z/OS MVS Programming: Assembler Services Reference, Volume 1 (ABEND-HSPSERV) and identify the cause of the GETMAIN error.
If the region size specified by the REGION parameter on the JOB or EXEC statement is not large enough, increase the size, and rerun the job.
FABM messages

Messages that are issued by the DBD/PSB/ACB Mapper utility begin with the prefix FABM.

FABM0001I  CONTROL CARD SUPPLIED IS: echo of control statement

Explanation: This message is the echo of the SYSIN control statements that this utility checks.

System action: The DBD/PSB/ACB Mapper utility continues processing.

User response: None. This message is informational.

FABM0002I  dbdname SELECTED

Explanation: The DBD dbdname has been found in the DBD library, and a map has been created.

System action: The DBD/PSB/ACB Mapper utility continues processing.

User response: None. This message is informational.

FABM0003I  dbdname SELECTED, EXTENDED REPORT

Explanation: The DBD dbdname has been found in the DBD library, and both a map and detailed descriptive report have been created.

System action: The DBD/PSB/ACB Mapper utility continues processing.

User response: None. This message is informational.

FABM0004W  NO MEMBER FOUND FOR dbdname IN DBDLIB

Explanation: The members, specified either specifically or by a wildcard, were not found in the DBD library. The indicated dbdname is the specified DBD member name or the applicable DBD member name specified by a wildcard.

System action: The DBD/PSB/ACB Mapper utility continues processing without printing the report for the member.

User response: Determine whether the specified dbdname is correct. If it is, search the library that has a member to be processed. Correct the problem, and rerun the job.

FABM0005I  dbdname DATA BASE HAS NO SEGMENTS

Explanation: The current DBD has no defined segments. No map is produced, and the report, if any, contains only database and data set group information.

System action: The DBD/PSB/ACB Mapper utility continues processing.

User response: Determine whether the specified dbdname is correct. If it is, search the library that has a member to be processed. Correct the problem, and rerun the job.

FABM0006E  dbdname segname SEGMENT NOT IN DATA BASE

Explanation: The indicated segment is described as a source found segment in the current DBD. However, it could not be found within its own DBD (dbdname).

System action: The DBD/PSB/ACB Mapper utility continues processing.

User response: Determine whether segname or dbdname is correct, and correct the problem. If the segment is described incorrectly in the current DBD, change the SEGM statement. Rerun the job.

FABM0008I  psbname SELECTED

Explanation: The PSB psbname has been found in the PSBLIB library. A PSB summary report and PSB maps (of DBDs specified in PCB statements) have been created.

System action: The DBD/PSB/ACB Mapper utility continues processing.

User response: None. This message is informational.

FABM0009I  psbname SELECTED, EXTENDED REPORT

Explanation: The PSB psbname has been found in the PSBLIB library. A PSB summary report, PSB maps, and PSB reports have been created.

System action: The DBD/PSB/ACB Mapper utility continues processing.

User response: None. This message is informational.

FABM0010W  NO MEMBER FOUND FOR psbname IN PSBLIB

Explanation: The members, specified either specifically or by a wildcard, were not found in the PSB library. The indicated psbname is the specified PSB member name or the applicable PSB member name specified by a wildcard.

System action: The DBD/PSB/ACB Mapper utility continues processing without printing the report for the member.

User response: Determine whether the specified psbname is correct. If it is, search the library that has a member to be processed. Correct the problem, and rerun the job.
FABM0011E  dbdname PCB ADDRESS PROBLEM
Explanation: The PSB has a PCB for the database dbdname that has an invalid address.
System action: Processing stops on this PSB and continues with the next PSB.
User response: The PSB is probably in error. Correct, reassemble, and link-edit the PSB; rerun the job.

FABM0012I  NO DB PCB FOUND IN PSB psbname
Explanation: There was no database-type PCB in the current PSB psbname.
System action: The DBD/PSB/ACB Mapper utility continues processing.
User response: None. This message is informational.

FABM0013E  dbdname DATA BASE HAS NO SEGMENTS
Explanation: The indicated DBD that the current PCB points to has no segments.
System action: Processing continues with the next PCB in the current PSB.
User response: Determine whether the error is in the DBD or in the current PSB. Correct the problem, and rerun the job.

FABM0014E  dbdname segname SEGMENT NOT IN DBD
Explanation: The indicated segment is described as a source segment in the database DBD that the current PCB points to. However, it could not be found in the DBD (dbdname).
System action: The DBD/PSB/ACB Mapper utility continues processing.
User response: Determine whether the error is in the database dbdname or in the current PCB. Correct the problem, and rerun the job.

FABM0015I  dbdname COMPACTION ERROR
Explanation: An error occurred in eliminating nonsensitive segments from the DBD dbdname. Processing was ended for the current PCB and continues to the next.
System action: The DBD/PSB/ACB Mapper utility continues processing.
User response: The DBD dbdname probably has an error. Correct, reassemble, and link-edit the DBD; rerun the job.

FABM0016W INVALID STATEMENT IN SYSIN DATASET
Explanation: A control statement with an invalid format was found in the SYSIN data set.
System action: The DBD/PSB/ACB Mapper utility skips this control statement and continues processing.
User response: Correct the format of this control statement, and rerun the job.

FABM0017E  LIBRARY MISSING
Explanation: The specifications of the libraries that are required to execute the function are missing or invalid.
System action: The DBD/PSB/ACB Mapper utility skips the reporting process for this function.
User response: Determine whether the required libraries are specified in the DD statements. Correct the DD statements for load module libraries, and rerun the job.

FABM0018I  acbname SELECTED, EXTENDED REPORT
Explanation: The PSB-type or DBD-type ACB acbname has been found in the ACB library. If it is a PSB-type, an ACB (PSB) summary report, ACB (PSB) maps, and ACB (PSB) reports have been created; otherwise an ACB (DBD) map and ACB (DBD) report have been created.
System action: The DBD/PSB/ACB Mapper utility continues processing.
User response: None. This message is informational.

FABM0019W NO MEMBER FOUND FOR acbname IN ACBLIB
Explanation: The members, specified either specifically or by a wildcard, were not found in the ACB library. The indicated acbname is the specified ACB member name or an applicable ACB member name specified by a wildcard.
System action: The DBD/PSB/ACB Mapper utility continues processing without printing the report of the member.
User response: Determine whether the specified acbname is correct. If it is, search the library that has a member to be processed. Correct the problem, and rerun the job.

FABM0020W member WAS NOT PSB TYPE ACB MEMBER
Explanation: The member specified is not a PSB-type ACB member. member is the member name that was specified in the SYSIN card or specified by a wildcard.
System action: The DBD/PSB/ACB Mapper utility skips this control statement and continues processing.

User response: None.

FABM0021E GETMAIN FAILED
Explanation: The program could not obtain sufficient area with the GETMAIN macro.
System action: The DBD/PSB/ACB Mapper utility ends abnormally.
User response: If the region is too small, increase it in the JOB statement in the JCL, and rerun the utility.

FABM0022E LINK FAILED FOR [DBD | PSB | ACB] MAP, RC=nnn
Explanation: An error occurred while a LINK macro was being issued during the DBD, PSB, or ACB map process. nnn is the system abend code.
System action: The DBD/PSB/ACB Mapper utility ends abnormally.
User response: Check the OS return code and determine the cause of the LINK macro failure, correct it, and rerun the utility.

FABM0023E ERROR READING ACB=member
Explanation: An error occurred while an ACB member was being read.
System action: The DBD/PSB/ACB Mapper utility continues processing without reporting this ACB.
User response: Determine the cause of the read error, correct it, and rerun the utility.

FABM0024E BLDL FAILED FOR ACB DIRECTORY
Explanation: An error occurred while a BLDL macro was being issued.
System action: The DBD/PSB/ACB Mapper utility skips the reporting process for the ACB Map function.
User response: Determine the cause of the BLDL macro failure, correct it, and rerun the utility.

FABM0025E UNSUPPORTED VERSION, ACB=x.x
Explanation: The ACB member generated by IMS version x.x is not supported.
System action: The DBD/PSB/ACB Mapper utility continues processing without reporting for this ACB member.
User response: Check the ACB members and the ACB library version.

FABM0026E SYSSOUT DID NOT OPEN
Explanation: The SYSSOUT data set could not be opened during initialization.
System action: The DBD/PSB/ACB Mapper utility ends abnormally.
User response: Determine the cause of the open failure.

FABM0027E SYSIN DID NOT OPEN
Explanation: The SYSIN data set could not be opened during initialization.
System action: The DBD/PSB/ACB Mapper utility ends abnormally.
User response: Determine the cause of the open failure.

FABM0028E SYSPRINT DID NOT OPEN
Explanation: The SYSPRINT data set could not be opened during initialization.
System action: The DBD/PSB/ACB Mapper utility ends abnormally.
User response: Determine the cause of the open failure.

FABM0029W MAXIMUM SYSIN CARDS EXCEEDED
Explanation: The number of control statements has reached the maximum value of 9999.
System action: Processes the first 9999 statements and ignores the rest.
User response: Rerun the ignored card.

FABM0030E ERROR LOADING [DBD | PSB] NAMED member (ABEND CODE=nnnn REASON CODE=mmmm)
Explanation: An error occurred while the DBD or the PSB member was being loaded. nnnn is the system abend code, and mmmm is its reason code.
System action: Skips this member and tries to load the next member if there is one.
User response: Determine the cause of load error. Correct the problem, and rerun the utility.

FABM0031W [DBD | PSB] member IS NOT A VALID [DBD | PSB]. ERROR IS DETECTED IN xxxxxxx
Explanation: DBD/PSB xxxxxxx was loaded, but was found not to be valid. If the invalid block can be identified, the block name follows.
System action: Skips this member and tries to load
The next member if there is one.

**User response:** Determine whether the member is a DBD or a PSB. If the member is a DBD or a PSB, regenerate the DBD or the PSB. If the member is not a DBD or a PSB, ignore this message.

**FABM0032E CONFLICT OF IMS VERSION DETECTED IN ACBLIB. ACB pppppppp IS IMS x.x, ACB dddddddd IS IMS y.y**

**Explanation:** PSB-type ACB pppppppp is generated by IMS version x.x, but the DBD-type ACB dddddddd referred to by pppppppp is generated by IMS version y.y. Both ACBs must be generated by the same version and release of IMS.

**System action:** The DBD/PSB/ACB Mapper utility continues processing without reporting for this ACB member.

**User response:** Check the ACB members and the ACB library version.

**FABM0033W fieldname IS NOT SPECIFIED AS SYSTEM RELATED FIELD IN [DBD dbdname | ACB acbname]**

**Explanation:** /CK and /SX are system-related field names. But the XDFLD statement that refers to fieldname was not found in DBD dbdname or in ACB acbname.

**System action:** The DBD/PSB/ACB Mapper utility continues processing.

**User response:** None.

**FABM0034I MEMBER member PROCESSED**

**Explanation:** This message is the echo of the member to be processed. member is the name of the member to be processed.

**System action:** The DBD/PSB/ACB Mapper utility continues processing.

**User response:** None. This message is informational.

**FABM0035E READ ERROR ON xxx DIRECTORY**

**Explanation:** A read error occurred while the directory was being read. xxx is one of DBD, PSB, or ACB.

**System action:** The DBD/PSB/ACB Mapper utility skips this function, issues return code 8, and continues processing.

**User response:** None.

**FABM0036W NO DATA IN SYSIN**

**Explanation:** There is no valid record in the SYSIN data set.

**System action:** The DBD/PSB/ACB Mapper utility ends without producing a map or a report.

**User response:** Specify a valid control record in the SYSIN data set, and rerun the utility.

**FABM0037W NO MEMBER NAME IS SPECIFIED**

**Explanation:** No member name is specified in the control statement.

**System action:** The DBD/PSB/ACB Mapper utility skips this control statement and continues processing.

**User response:** Specify a valid member name, and rerun the utility.

**FABM0038W INVALID MEMBER NAME IS SPECIFIED**

**Explanation:** The member name specified in the control statement is incorrect. For example, the name is longer than eight characters.

**System action:** The DBD/PSB/ACB Mapper utility skips this control record and continues processing.

**User response:** Specify a valid member name, and rerun the utility.

**FABM0039I acbname SELECTED**

**Explanation:** The PSB-type or DBD-type ACB acbname has been found in the ACB library. If it is of PSB-type, an ACB (PSB) summary report and ACB (PSB) maps have been created; otherwise an ACB (DBD) map has been created.

**System action:** The DBD/PSB/ACB Mapper utility continues processing.

**User response:** None. This message is informational.

**FABM0040W acbname WAS NOT DBD TYPE ACB MEMBER**

**Explanation:** The members, specified either specifically or by a wildcard, were not DBD-type ACB members. The indicated acbname is the specified member name or applicable member name specified by a wildcard.

**System action:** The DBD/PSB/ACB Mapper utility skips this control statement and continues processing.

**User response:** None.
FABM0041W SEGMENT $FABMnnn ON DBD
dbdname IS NAMED BY DBD/PSB/ACB
MAPPER AUTOMATICALLY

Explanation: DBD segment name was assigned by the
DBD/PSB/ACB Mapper utility automatically because
the name could not be obtained from any ACB
members. $FABMnnn is the name assigned; nnn shows
the segment code; and dbdname is the DBD name that
was decoded.

System action: The DBD/PSB/ACB Mapper utility
continues processing.

User response: None.

FABM0042E SEGMENT 1st-segname ON DBD
dbdname IS DEFINED AS SEGMENT
2nd-segname ON PSB psbname

Explanation: A mismatch of segment names between
DBD dbdname and PSB psbname was found while
processing the DBD-type ACB member. 1st-segname is
the correct segment name on the DBD dbdname,
however, the PSB-type ACB psbname, which was
decoded before the DBD, referred to the segment by
2nd-segname.

System action: The DBD/PSB/ACB Mapper utility
continues processing.

User response: None.

FABM0043W REAL LOGICAL CHILD SEGMENT
$SEGMnnn AND DBD NAME
$DBDnnn ON DBD dbdname ARE
NAMED BY DBD/PSB/ACB MAPPER
AUTOMATICALLY

Explanation: Real logical child segment name
$SEGMnnn and DBD name $DBDnnn in which the
segment resides were assigned by the DBD/PSB/ACB
Mapper utility automatically, because the names could
not be obtained from the specified ACB member. nnn is
a sequential number in a DBD, and dbdname is the
name of the DBD that was processed.

System action: The DBD/PSB/ACB Mapper utility
continues processing.

User response: None.

FABM0044W UNSUPPORTED DBD member IN
DBDLIB

Explanation: The specified DBD member in the DBD
library is not supported for the DBD/PSB/ACB
Mapper utility.

System action: The DBD/PSB/ACB Mapper utility
skips this member and continues processing.

User response: Check the DBD member.
## FABN messages

Messages that are issued by the DBD/PSB/ACB Reversal utility and the Reversal Site Default Generation utility begin with the prefix FABN.

<table>
<thead>
<tr>
<th>FABN0001E</th>
<th>SYSIN DID NOT OPEN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>The SYSIN data set could not be opened during initialization.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility ends abnormally.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>Determine the cause of the open failure. Correct the problem, and rerun the program.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0003E</th>
<th>SYOUT DID NOT OPEN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>The SYOUT data set could not be opened during initialization.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility ends abnormally.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>Determine the cause of the open failure. Correct the problem, and rerun the program.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0004E</th>
<th>FABNRVRS MAXIMUM SAVEAREA COUNT EXCEEDED</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>The save area count exceeds the maximum count for this utility.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility ends abnormally.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>Determine the cause of the error. Correct the problem, and rerun the program.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0005E</th>
<th>FABNRVRS NO MORE VIRTUAL STORAGE AVAILABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>The program could not obtain sufficient area with the GETMAIN macro.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility ends abnormally.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>Increase the region size in the JOB statement in the JCL, and rerun the utility.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0006I</th>
<th>BLANK RECORD IN SYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>There is a blank record in the SYSIN data set.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility continues processing.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>None. This message is informational.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0007E</th>
<th>UNKNOWN KEYWORD IN SYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>An unknown keyword is found in the SYSIN data set.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility ends with a return code of 16.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>Correct the keyword, and rerun the utility.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0008W</th>
<th>UNKNOWN OPERAND IN SYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>There is an unknown operand in the SYSIN data set.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility skips this record and continues processing.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>Correct the operand, and rerun the utility.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0009W</th>
<th>UNKNOWN OPTION IN SYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>There is an unknown option in the SYSIN data set.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility skips this record and continues processing.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>Correct the option, and rerun the utility.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0010W</th>
<th>INVALID RECORD IN SYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>There is a record with an invalid format in the SYSIN data set.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility skips this record and continues processing.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>Correct the SYSIN record, and rerun the utility.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0011W</th>
<th>NO DATA IN SYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>There is no valid record in the SYSIN data set.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>The DBD/PSB/ACB Reversal utility ends without producing reports or source codes.</td>
</tr>
<tr>
<td><strong>User response:</strong></td>
<td>Specify valid control records in the SYSIN data set, and rerun the utility.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FABN0012E</th>
<th>DBD NAMED [dbdname 1 acbname] HAS UNKNOWN ACCESS CODE - HEX nn</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>The loaded DBD (or DBD-type ACB) dbdname has an unknown access method code. nn is the unknown code, in hexadecimal.</td>
</tr>
<tr>
<td><strong>System action:</strong></td>
<td>Ends the reversal process for this DBD and issues an error message with dump of this module to the SYOUT data set. Then continues processing with the next module.</td>
</tr>
</tbody>
</table>
User response: Correct the DBD having the unknown access code, and rerun the utility.

FABN0013E  [DBD | PSB] NAMED member IS MARKED NOT EXECUTABLE; CANNOT BE LOADED

Explanation: The member member in the DBD/PSB load library was marked 'not executable' before it was loaded.

System action: Skips this member and tries to load the next member if it exists.

User response: Correct the error of this module, and generate this member (DBDGEN/PSBGEN) again. Then rerun the utility.

FABN0014E ERROR LOADING [DBD | PSB] NAMED member (ABEND CODE=nnnn REASON CODE=nnnnn)

Explanation: An error occurred during DBD/PSB member load. nnnn is the system abend code, and nnnnn is its reason code.

System action: Skips this member and tries to load the next member if it exists.

User response: Determine the cause of the load error. Correct the problem, and rerun the utility.

FABN0015W NO MEMBER FOUND FOR member IN [DBDLIB | PSBLIB | ACBLIB]

Explanation: The member specified by the EXCLUDE= or INCLUDE= option or by a wildcard was not found in the DBD/PSB/ACB library. The member is the name of the DBD/PSB/ACB having the unknown type option, and member names specified by a wildcard.

System action: If a member is specified in the INCLUDE= option, the DBD/PSB/ACB Reversal utility skips processing this member. If a member is specified in the EXCLUDE= option, the DBD/PSB/ACB Reversal utility attempts to process all members in the library.

User response: Determine if member is correct. If it is, search the library that contains it. After determining the cause of the failure, rerun the utility.

FABN0016E UNABLE TO LOCATE [DBD | PSB] NAMED member

Explanation: The member specified by the INCLUDE= option was not found in the DBD/PSB load library.

System action: Skips this member and then tries to load the next member if it exists.

User response: Correct the member name, and rerun the utility, if necessary.

FABN0017E UNKNOWN TYPE OPTION X'nnn' FOR FIELD fieldname IN SEGMENT segname IN DBD [dbdname | acbname]

Explanation: The DBD has an unknown type option in the FIELD statement. X'nnn' is the unknown type option in hexadecimal, fieldname is the FIELD name, segname is the SEGMENT name, and ddbname is the DBD name.

System action: Stops processing this module and writes an error message with dump for this module in the SYSOUT data set. Then continues processing the next module.

User response: Correct the type option, and rerun the utility, if necessary.

FABN0018E INVALID TYPE OPTION - SEQ - IN SEGMENT segname IN DBD [dbdname | acbname]

Explanation: The segment type option TYPE=SEQ is invalid in a DEDB database. It must be the first dependent segment type, and only one such segment type is allowed for the DEDB database. segname is the name of the segment that has an incorrect SEQ-type option, and ddbname is the DEDB database name.

System action: Stops processing this DBD, and outputs an error message with dump of this DBD to the SYSOUT data set. Then continues processing the next DBD.

User response: Correct the segment type option and, if necessary, rerun the utility.

FABN0019E MEMBER NAME (xxxxxxxx) AND [DBD | PSB] NAME (yyyyyyyy) MISMATCH

Explanation: The member name of the DBD does not match the DBD/PSB name in which it is defined. xxxxxxxxx is the DBD/PSB with a mismatched member name, and yyyyyyyyy is the DBD/PSB name.

System action: Stops processing this DBD, outputs an error message with partial dump of this DBD to the SYSOUT data set, and continues processing the next module.

User response: Correct the member name or DBD/PSB name so that they match, and rerun the utility.

FABN0020W NO MEMBER NAME IS SPECIFIED

Explanation: No member name is specified for the INCLUDE or the EXCLUDE option.

System action: The DBD/PSB/ACB Reversal utility skips this control statement and continues processing.

User response: Specify a valid member name, and rerun the utility.
FABN0021W  INVALID MEMBER NAME IS SPECIFIED

Explanation: The member name specified for the INCLUDE or the EXCLUDE option is incorrect. For example, the name contains more than eight characters, or is delimited by more than one comma.

System action: The DBD/PSB/ACB Reversal utility skips this control record and continues processing.

User response: Specify a valid member name, and rerun the utility.

FABN0022E  NO DATASET GROUP FOUND FOR SEGMENT segname IN DBD [dbdname | acbname] DSID id

Explanation: The data set name that is specified in the SEGMENT statement is not found in the DATASET statement in this DBD. segname is the name of the SEGMENT that has an incorrect data set group ID, dbdname is the DBD name, and id is the invalid data set ID.

System action: Stops processing this DBD and outputs an error message with partition dump of this DBD to the SYSOUT data set. Then continues processing the next DBD.

User response: Correct the data set group ID, and rerun the utility if necessary.

FABN0023E  DD STATEMENT xxxxxxx DID NOT OPEN

Explanation: The DD statement could not be opened. xxxxxxx is the DD statement name.

System action: The DBD/PSB/ACB Reversal utility skips the process related to this DD statement.

User response: Determine the cause of the open error. Correct the error, and rerun the utility.

FABN0024I  CONTROL CARD SUPPLIED IS: echo of control statement

Explanation: This message is the echo of control statements in the SYSIN data set.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0025E  READ ERROR ON [DBD | PSB | ACB | DBDSRC | PSBSRC] DIRECTORY

Explanation: An error occurred while the directory was being read.

System action: The DBD/PSB/ACB Reversal utility skips this function.

User response: Determine the cause of READ error. Correct the problem, and rerun the utility.

FABN0026I  MAPOUT FUNCTION [SELECTED | NOT SELECTED]

Explanation: This message tells whether the Mapper input generation function has been selected. When the MAPOUT data set is successfully opened, this utility issues a Mapper input generation function as a subfunction of the DBD/PSB/ACB Reversal function or the DBD/PSB SEGMENT reference report function. If the MAPOUT data set is not specified in the JCL or cannot be opened, this utility assumes that the mapper input generation function has not been selected.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0027I  NO DBD TO [DBD | DBD TYPE ACB] TO [DBD | DBD TYPE ACB] REFERENCE FOUND

Explanation: DBD is specified on the XREF control statement, but no DBD references to other DBDs are found. Neither the Reference report nor the Referenced report of the DBD to DBD Xref report (or the ACB(DBD) to ACB(DBD) Xref report) is printed.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0028I  NO PSB TO [DBD | PSB TYPE ACB] TO [DBD | DBD TYPE ACB] REFERENCE FOUND

Explanation: PSB is specified on the XREF control statement, but no DBD references to DB/GSAM PCBs are found. Neither the Reference report nor the Referenced report of the DBD to DBD Xref report (or the ACB(DBD) to ACB(DBD) Xref report) is printed.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0029I  NO REFERENCED [DBD | DBD TYPE ACB] FOUND

Explanation: No DBD is referred to in other DBDs in the DBDLIB (or ACBLIB). The Referenced report of the DBD to DBD Xref report (or the ACB(DBD) to ACB(DBD) Xref report) is not printed.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.
FABN0030W  MACRO OPERAND EXCEEDS ASSEMBLER LIMITATION - member, LINE=nnn

Explanation: The MACRO operand length decoded by the DBD/PSB/ACB Reversal utility exceeds the Assembler limitation. If the user runs the IMS DBDGEN or the PSBGEN utility with the decoded MACRO source statements as input, an Assembler error will occur. member is the DBD or the PSB name, and nnn is the line number on which the output operand length exceeds the limitation.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: The DBD/PSB/ACB Reversal utility decodes the MACRO source and generates source statements with full operands explicitly coded. Delete the default operand values from the source statements to reduce the operand length, if necessary, and run the IMS DBDGEN/PSBGEN utility, using the modified source statements.

FABN0031W  [DBD | PSB] member IS NOT A VALID [DBD | PSB]. ERROR IS DETECTED IN xxxxxxx

Explanation: DBD/PSB xxxxxxxxx was loaded, but was found not to be valid. If the invalid block can be identified, the block name follows.

System action: Skips this member and tries to load the next member if it exists.

User response: Determine if the member is a DBD or a PSB. If the member is a DBD or a PSB, regenerate the member. If it is not, ignore this message.

FABN0032I  MEMBER member PROCESSED

Explanation: The member member has been processed.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0033W  NO MEMBER TO BE PROCESSED FOR CONTROL CARD

Explanation: No member specified in the EXCLUDE= option has been processed.

System action: None.

User response: Determine whether the control statement is correct. If the control statement is correct, search the library that has a member to be processed. Determine the cause, and rerun.

FABN0034E  GETMAIN FAILED

Explanation: The program could not obtain enough area with GETMAIN macro.

System action: The DBD/PSB/ACB Reversal utility ends abnormally.

User response: If the specified region size is too small, increase the REGION size in the JOB statement in the JCL, and rerun the utility.

FABN0035E  ERROR READING ACB NAMED acbname (ABEND CODE=nnnn REASON CODE=mmm)

Explanation: An error occurred while an ACB member was being read. nnnn is the system abend code, and mmm is its reason code.

System action: The DBD/PSB/ACB Reversal utility skips this control statement and continues processing.

User response: Check the system abend code and its reason code. Determine the cause of the read error, correct it, and rerun the utility.

FABN0036E  BLDL FAILED FOR [ACB | PSB] DIRECTORY

Explanation: An error occurred while a BLDL macro was being issued for the ACB or the PSB directory.

System action: The DBD/PSB/ACB Reversal utility skips this control statement and continues processing.

User response: Determine the cause of the BLDL macro failure, correct it, and rerun the utility.

FABN0037W  member WAS NOT PSB TYPE ACB MEMBER

Explanation: The member specified is not a PSB-type ACB member. member is the member name that was specified in the SYSIN card or specified by a wildcard.

System action: The DBD/PSB/ACB Reversal utility skips this control statement and continues processing.

User response: None.

FABN0038E  ACB(DBD) member REFERRED BY acbname NOT FOUND IN ACBLIB

Explanation: The DBD-type ACB member referred to by the PSB-type ACB member specified in the SYSIN card was not found in the ACB load library. member is the DBD-type ACB member name, and acbname is the PSB-type ACB member name specified in the SYSIN card.

System action: The DBD/PSB/ACB Reversal utility skips this control statement and continues processing.

User response: None.
FABN0039W  SEGMENT NAME $FABNmmu ON DBD
dbdname IS NAMED BY DBD/PSB/ACB REVERSAL AUTOMATICALLY

Explanation:  DBD segment name was assigned by the DBD/PSB/ACB Reversal utility automatically, because the name could not be obtained from any ACB members. $FABNmmu is the name assigned, mnu shows the segment code, and dbdname is the name of the DBD that was decoded.

System action:  The DBD/PSB/ACB Reversal utility continues processing.

User response:  None.

FABN0040E  SEGMENT 1st-segname ON DBD
dbdname IS DEFINED AS SEGMENT
2nd-segname ON PSB psbname

Explanation:  A mismatch of segment names between DBD dbdname and PSB psbname was found while processing the DBD-type ACB member. 1st-segname is the correct segment name on the DBD dbdname, however, the PSB-type ACB psbname, which was decoded before the DBD, referred to the segment by 2nd-segname.

System action:  The DBD/PSB/ACB Reversal utility continues processing.

User response:  None.

FABN0041E  UNSUPPORTED VERSION, ACB=xx

Explanation:  This message is issued if the IMS version of PSB-type ACB member in the ACB library is not supported.  ACB=xx is the IMS version of PSB-type ACB member in the ACB library.

System action:  The DBD/PSB/ACB Reversal utility skips this control statement and continues processing.

User response:  Check the ACB member and the ACB library version.

FABN0042E  CONFLICT OF IMS VERSION
DETECTED IN ACBLIB.  ACB member1 IS IMS xx.xx, ACB member2 IS IMS yy.yy

Explanation:  PSB-type ACB or DBD-type ACB (member1) is generated by IMS version xx.xx, but the DBD-type ACB (member2) referred to by the PSB-type ACB or DBD-type ACB (member1) is generated by IMS version yy.yy.  Both ACBs must be generated by the same version and release of IMS.

System action:  The DBD/PSB/ACB Reversal utility skips this control statement and continues processing.

User response:  Check the ACB members and the ACB library version.

FABN0043I  DECODED MEMBER member WRITTEN TO [DBDSRC | PSBSRC]

Explanation:  An IMS DBDGEN/PSBGEN utility control statement of a decoded member was written to PDS or PDSE data set.

System action:  The DBD/PSB/ACB Reversal utility continues processing.

User response:  None.  This message is informational.

FABN0044I  member IN [DBDSRC | PSBSRC] REPLACED

Explanation:  An IMS DBDGEN/PSBGEN utility control statement was written to the PDS or PDSE data set by replacing it with a new member, member, because the same member name already exists.

System action:  The DBD/PSB/ACB Reversal utility continues processing.

User response:  None.  This message is informational.

FABN0045E  WRITE ERROR ON [DBDSRC | PSBSRC], DECODED MEMBER = member

Explanation:  An error occurred while a decoded IMS DBDGEN/PSBGEN utility control statement was being written.  member is the name of the member that was being written.

System action:  The DBD/PSB/ACB Reversal utility continues processing.

User response:  Correct the problem, and rerun the job.

FABN0046E  STOW ERROR ON [DBDSRC | PSBSRC] DIRECTORY, DECODED MEMBER = member (RETURN CODE = mmm, REASON CODE = mmmm)

Explanation:  An error occurred while a PDS or PDSE directory entry of member member was being added, updated, or deleted in the DBDSRC or the PSBSRC data set.  mmm is the system abend code, and mmmm is the reason code.

System action:  The DBD/PSB/ACB Reversal utility ends abnormally.

User response:  Determine the cause of the error.  For the return codes and the reason codes, see the topic "STOW completion codes" in z/OS DFSMS Macro Instructions for Data Sets. Corract the problem, and rerun the job.
FABN0047E  member IN [DBDSRC | PSBSRC] NOT FOUND

Explanation: The member name member was not found when the DBD/PSB/ACB Reversal utility deleted or updated a member in the PDS/PDSE data set.

System action: The DBD/PSB/ACB Reversal utility does not delete or update the member. The DBD/PSB/ACB Reversal utility continues processing.

User response: None.

FABN0048W  GSAM PCB (NUM=nnn) IS NOT DECODED IN PSB acbname

Explanation: The DBD/PSB/ACB Reversal program did not decode the GSAM PCB. acbname is the PSB-type ACB member name that was specified, and nnn is the number of GSAM PCB that were specified. If NUM=N/A is shown, it indicates that the DBD/PSB/ACB Reversal program could get no information about the number of GSAM PCBs.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None.

FABN0049I  NO DB PCB FOUND IN PSB acbname

Explanation: There was no DB PCB in the current PSB-type ACB acbname.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0050W  NO SEGMENT NAME IS SPECIFIED

Explanation: No segment name is specified for the SEGMENT option.

System action: The DBD/PSB/ACB Reversal utility skips this control statement and continues processing.

User response: Specify a valid segment name, and rerun the utility.

FABN0051W  INVALID SEGMENT NAME IS SPECIFIED

Explanation: The segment name has an invalid format. For example, the name is longer than eight characters, or the first character of the segment name in the SYSIN data set is a comma.

System action: The DBD/PSB/ACB Reversal utility skips this control statement and continues processing.

User response: Specify a valid segment name, and rerun the utility.

FABN0052I  SEGMENT NAME segname PROCESSED

Explanation: The segment name segname is processed.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0053W  NO SEGMENT NAME FOUND FOR SPECIFIED segname IN SEARCHED MEMBER(S) IN xxx LIB

Explanation: The segment name specified by the SEGNAME=segname or by a wildcard was not found in searched members in the DBD/PSB library. segname is the segment name or the segment name specified by a wildcard, and xxx is DBD or PSB.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: Correct the segment name and, if necessary, rerun the utility.

FABN0054I  MEMBER member HAS NO DDNAME

Explanation: A DBD member of HALDB was specified for the DDNAMES keyword. The DBD/PSB/ACB Reversal program skips this record, because the DBD has no DD name.

System action: The DBD/PSB/ACB Reversal utility skips this record and continues processing.

User response: None. This message is informational.

FABN0055W  DBD NAME $DBD nnn ON PSB psbname IS NAMED BY DBD/PSB/ACB REVERSAL AUTOMATICALLY

Explanation: PSB psbname was decoded, and a DBD name was assigned by the DBD/PSB/ACB Reversal utility automatically, because the name could not be obtained from a PSB-type ACB member. $DBDnnn is the name assigned; nnn is the sequence number in a PSB.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None.

FABN0056E  A SEGMENT REFERRED TO BY [DBD | PSB] member IS NOT FOUND IN DBD dbdname, DBD/PSB/ACB REVERSAL NAMED IT $FABN nnn AUTOMATICALLY

Explanation: DBD/PSB member referred to a segment in DBD dbdname, but the segment was not found in the DBD. The segment name was assigned by the DBD/PSB/ACB Reversal utility automatically in DBD/PSB member, because the name could not be
obtained from any ACB members. $FABNm$ is the name assigned; $nnn$ shows the segment code.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None.

FABN0057W  $acbname$ WAS NOT DBD TYPE ACB MEMBER

Explanation: The specified member was not a DBD-type ACB member. The specified $acbname$ is the member name that was specified in the SYSIN card or specified by a wildcard.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None.

FABN0058W  NO PCBNAME PREFIX IS SPECIFIED

Explanation: Prefixes for PCB names are not specified for the PCBNAMEX option.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: Specify an appropriate prefix for the PCB names, and rerun the utility.

FABN0059W  INCORRECT PCBNAME PREFIX IS SPECIFIED

Explanation: The PCBNAME prefix specified for the PCBNAMEX option has an incorrect format. For example, the prefix is longer than four characters.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: Specify an appropriate PCBNAME prefix, and rerun the utility.

FABN0060I  PCBNAMEX=prfx option is used

Explanation: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0061I  PCBNAME prfxnnnn on PSB $psbname$ is named by DBD/PSB/ACB Reversal automatically

Explanation: PSB $psbname$ was decoded and the PCB name $prfxnnnn$ was assigned by the DBD/PSB/ACB Reversal utility automatically, because the PCB name had not been defined and the PCBNAMEX option with the DECODE keyword was specified for the PSB. $prfx$ is the prefix specified by the PCBNAMEX= option and $nnnn$ is the sequence number for the PCB in the PSB, which is equal to the PCB number.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0062I  keyword USED IS: $xxxxxx$

Explanation: The DBD/PSB/ACB Reversal utility proceeds with the indicated runtime option.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0063I  DBDNAME=xxxxxx option is used

Explanation: The DBD/PSB/ACB Reversal utility proceeds with the specified DBDNAME option.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0064W  NO PSB REFERRING TO $dbdname$ IS FOUND IN $xxaLIB$

Explanation: There is no PSB or PSB-type ACB that refers to the DBD $dbdname$ specified by the DBDNAME option in the PSBLIB or ACBLIB. The $dbdname$ can be specified with its exact name or with the use of wildcards.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: Correct the $dbdname$, and rerun the utility if necessary.

FABN0065W  REAL LOGICAL CHILD SEGMENT $SEGNm$ AND DBD NAME $DBDnnn$ ON DBD $dbdname$ ARE NAMED BY DBD/PSB/ACB REVERSAL AUTOMATICALLY

Explanation: Real logical child segment name $SEGNm$ and DBD name $DBDnnn$ in which the segment resides were assigned by the DBD/PSB/ACB Reversal utility automatically, because the names could not be obtained from the specified ACB member. $nnn$ is a sequential number in a DBD, and $dbdname$ is the name of the DBD that was decoded.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None.
FABN0066W UNSUPPORTED DBD member IN [DBDLIB | ACBLIB]

Explanation: The specified DBD member in the DBD or ACB library is not supported by the DBD/PSB/ACB Reversal utility.

System action: The DBD/PSB/ACB Reversal utility ignores this member and continues processing.

User response: Check the DBD member in the DBD or ACB library.

FABN0068W RDMVT CSECT IS CUSTOMIZED: MEMBER=member, TYPE=[DBD | DBD TYPE ACB]

Explanation: While processing member, the DBD/PSB/ACB Reversal utility detected one or more customized fields in RDMVT CSECT (described by the DMBDACS DSECT) that contain the randomizing information. One or more of the following fields are detected as customized by the DBD/PSB/ACB Reversal utility.

<table>
<thead>
<tr>
<th>Detectable Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMBDASZE</td>
<td>The size of RDMVTAB CSECT</td>
</tr>
<tr>
<td>DMBDAKL</td>
<td>The executable key length of root</td>
</tr>
<tr>
<td>DMBDANME</td>
<td>The name of randomizer module</td>
</tr>
<tr>
<td>DMBDARAP</td>
<td>The number of root anchor points or blocks</td>
</tr>
<tr>
<td>DMBDABLK</td>
<td>The number of the highest blocks that are directly addressed</td>
</tr>
<tr>
<td>DMBDABYM</td>
<td>The maximum number of bytes</td>
</tr>
</tbody>
</table>

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: Check the randomizing information in the generated source and make sure that the randomizing information is appropriate for use.

FABN0070I SITE DEFAULT TABLE FABNCTL0 IS USED

Explanation: The site default table module FABNCTL0 is used by the DBD/PSB/ACB Reversal utility.

System action: The DBD/PSB/ACB Reversal utility receives the options that are specified in the site default table and uses them as the default values for the SYSIN control card.

User response: None. This message is informational.

FABN0071I SITE DEFAULT OPTION USED: option

Explanation: The option indicated by option is registered in the site default table.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0072E SITE DEFAULT TABLE FABNCTL0 IS CORRUPTED

Explanation: The site default table module (FABNCTL0) is corrupted and site default values are not used.

System action: The FABNRVRS program of the DBD/PSB/ACB Reversal utility ends with a return code of 8.

User response: Specify the correct FABNCTL0 module and ensure that the first 8 bytes of the site default table shows FABNCTL0. If the module is damaged, re-create another site default table module and store it in the STEPLIB data set.

FABN0073E LOAD FAILED FOR DDNAME ddname MODULE module

Explanation: After a LOAD macro was issued to load the module module from the library that is specified by the ddname DD, register 15 contained a nonzero return code.

System action: The FABNRVRS program of the DBD/PSB/ACB Reversal utility ends with a return code of 8.

User response: Ensure that the DD statement specifies the appropriate data set. Correct the error and rerun the job. If the problem persists, save the entire run listing (including the dump, JCL, and all reports from the DBD/PSB/ACB Reversal utility), and contact IBM Software Support.

FABN0074I THE INFORMATION ABOUT GSAM [DBD | PCB] IS RETRIEVED FROM membername

Explanation: The DBD/PSB/ACB Reversal utility is attempting to obtain GSAM DBD information from a DBD member in the DBDLIB or GSAM PCB information from a PSB member in the PSBLIB.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.
FABN0075E  membername IS NOT CONSISTENT BETWEEN ACBLIB AND [DBDLIB | PSBLIB]

Explanation: The member in the ACBLIB is not consistent with the member that has the same name in the DBDLIB or PSBLIB.

System action: The DBD/PSB/ACB Reversal utility skips the member in the ACBLIB, and continues processing.

User response: Determine if the correct DBDLIB or PSBLIB is specified on the DD statement. If necessary, rerun the utility.

FABN0076E  THE keyword STATEMENT IS INCORRECT

Explanation: Either the option that is specified for the indicated keyword is incorrect or the order of the keywords in the SYSIN data set is incorrect.

System action: The DBD/PSB/ACB Reversal utility ends with a return code of 16.

User response: Ensure that the option specified for the indicated keyword is correct. If an OPTION statement is specified, ensure that the statement is located at the top of the SYSIN data set. Rerun the utility.

FABN0077W  VERSION PARAMETER DBD STATEMENT IS NOT DECODED.

Explanation: The following parameters are not decoded because these parameters could not be obtained from the DBD member:

- The VERSION parameter of the DBD statement
- The EXIT parameter of the DBD and SEGM statements

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None.

FABN0078W  PSB THAT REFERS TO DBD dbdname WITH PROCOPT procopt IS NOT FOUND IN THE [PSBLIB | ACBLIB]

Explanation: In the specified PSB library or ACB library, no PSBs match the processing option criteria and the DBD name criteria that were defined for the POPTREF keyword. The processing option criteria are specified by the SEARCHOPT option, and the DBD name criteria are specified by the SEARCHDBD option.

System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: If necessary, update the criteria by modifying the SEARCHOPT and SEARCHDBD options, and then rerun the utility.

FABN0079W  THE SEARCHOPT PARAMETER CONTAINS INVALID CHARACTERS: characters

Explanation: The value that is specified for the SEARCHOPT option is incorrect. The value might be incorrect because it contains letters that do not correspond to processing options (PROCOPT), the same letter is specified more than once, or wildcard characters are specified before letters.

System action: The DBD/PSB/ACB Reversal utility skips this control record and continues processing.

User response: For a list of the letters that can be specified as processing options (PROCOPT), see the topic "Full-function or Fast Path database PCB statement" in IMS System Utilities. Correct the value for the SEARCHOPT option and rerun the utility.

FABN0080W  THE [SEARCHDBD | SEARCHOPT] PARAMETER CONTAINS AN INVALID VALUE.

Explanation: The DBD name that is specified by the SEARCHDBD option is longer than eight characters, or the list of processing options (PROCOPT) that is specified by the SEARCHOPT option is more than four characters.

System action: The DBD/PSB/ACB Reversal utility skips this control record and continues processing.

User response: Correct the value that is specified for the SEARCHDBD option or the SEARCHOPT option, and rerun the utility.

FABN0081W  ONE OR MORE REQUIRED OPTIONS ARE MISSING OR THE ORDER IS INCORRECT.

Explanation: The required options are missing, or the order of the options is incorrect.

System action: The DBD/PSB/ACB Reversal utility skips this control record and continues processing.

User response: See "Control statements for the DBD/PSB/ACB Reversal utility" on page 288 and specify the required options or correct the order of the options, and then rerun the utility.

FABN0082I  DBD TYPE ACB MEMBER dbdname IS NOT REFERRED BY ANY PSB TYPE ACB MEMBERS

Explanation: The DBD/PSB/ACB Reversal utility found a DBD-type ACB member that is not referenced by any PSB-type ACB member in the ACBLIB.
System action: The DBD/PSB/ACB Reversal utility continues processing.

User response: None. This message is informational.

FABN0083W NO OPTIONS ALLOWED FOR control_statement

Explanation: Function keyword options, such as ALL, INCLUDE=, EXCLUDE=, and so on, are not supported for the UNREF ACB, LISTLIB DBD, and LISTLIB PSB statements.

System action: The DBD/PSB/ACB Reversal utility ignores the UNREF ACB, LISTLIB DBD, or LISTLIB PSB control statement and continues processing.

User response: Remove the function keyword options from the UNREF ACB, LISTLIB DBD, or LISTLIB PSB statement, and rerun the utility.

FABN0084W MEMBER member WAS NOT GENERATED BY THE IMS STANDARD DBD OR PSB GENERATION MACRO

Explanation: The indicated DBD or PSB member has a control block length that does not conform to the control block length of a DBD or PSB that is generated by the IMS DBDGEN or IMS PSBGEN utility. This member is assumed to be generated or modified by using a non-IMS macro.

System action: The DBD/PSB/ACB Reversal utility continues processing and ends with a return code of 04.

User response: None. This message is informational.

FABN0085E ACCESS FAILED FOR cataloghlq. FUNC=function, RETURN CODE=rc, REASON CODE=rsn

Explanation: IMS Library Integrity Utilities detected an error while accessing the IMS catalog directory.

System action: IMS Library Integrity Utilities ends with a return code of 8.

User response: Locate the GEX3xxxE message that is issued before this message. For the meaning of the GEX3xxxE message, see the topic "IMS Tools Catalog Interface messages (GEX3)" in the Tools Base IMS Tools Common Services V1.6 User’s Guide. If necessary, correct the error condition and rerun the job.

FABN0086W GSAM DBD dbdname IN IMS CATALOG IS NOT VALID.

Explanation: The indicated GSAM DBD is found in the IMS catalog directory but it is not in a valid format.

System action: Skips this member and tries to load the next member if it exists.

User response: To correct the DBD, run the DBD, PSB, and ACB generation utilities and then use the IMS Catalog Populate utility (DFS3PU00) to populate the IMS catalog. You can also use the ACB Generation and Catalog Populate utility (DFS3UACB) to generate and populate ACBs.

FABN1000I FABNTGEN ENDED NORMALLY

Explanation: The site default generation utility ended normally.

System action: The site default generation utility ends with a return code of 0.

User response: None. This message is informational.

FABN1004I SITE DEFAULT OPTION USED: option

Explanation: The option indicated by option is registered in the site default table.

System action: The site default generation utility continues processing.

User response: None. This message is informational.

FABN1006I BLANK RECORD IN SYSIN

Explanation: There is a blank record in the SYSIN data set.

System action: The site default generation utility continues processing.

User response: None. This message is informational.

FABN1020I THE SOURCE CODE FOR THE SITE DEFAULT TABLE IS GENERATED

Explanation: The source code for the site default table is generated by the FABNTGEN program.

System action: The site default generation utility continues processing.

User response: None. This message is informational.

FABN1021I CONTROL CARD SUPPLIED IS: control_statement

Explanation: This message shows the echo of control statements that are specified in the SYSIN data set.

System action: The site default generation utility continues processing.

User response: None. This message is informational.

FABN1022I KEYWORD keyword IS IGNORED

Explanation: The keyword indicated by keyword is ignored.

System action: The site default generation utility continues processing.
<table>
<thead>
<tr>
<th>Code</th>
<th>Message</th>
<th>Explanation</th>
<th>System action</th>
<th>User response</th>
</tr>
</thead>
<tbody>
<tr>
<td>FABN1023I</td>
<td>\textit{keyword} \texttt{OPTION} \texttt{USED}: option</td>
<td>The site default generation utility proceeds with the runtime option.</td>
<td>The site default generation utility continues processing.</td>
<td>None. This message is informational.</td>
</tr>
<tr>
<td></td>
<td><strong>Explanation</strong>: The site default generation utility proceeds with the runtime option.</td>
<td><strong>System action</strong>: The site default generation utility continues processing.</td>
<td><strong>User response</strong>: None. This message is informational.</td>
<td></td>
</tr>
<tr>
<td>FABN1030I</td>
<td>\texttt{SITE DEFAULT TABLE FABNCTL0} IS PRINTED</td>
<td>The specification in the site default table is reported by the FABNTGEN program.</td>
<td>The site default generation utility continues processing.</td>
<td>None. This message is informational.</td>
</tr>
<tr>
<td></td>
<td><strong>Explanation</strong>: The specification in the site default table is reported by the FABNTGEN program.</td>
<td><strong>System action</strong>: The site default generation utility continues processing.</td>
<td><strong>User response</strong>: None. This message is informational.</td>
<td></td>
</tr>
<tr>
<td>FABN1000W</td>
<td>\texttt{FABNTGEN ENDED WITH WARNINGS}</td>
<td>The site default generation utility ended with warning conditions.</td>
<td>The site default generation utility ends with a return code of 4.</td>
<td>Check the preceding messages that explain the warning conditions. Correct the warning conditions and rerun the utility.</td>
</tr>
<tr>
<td></td>
<td><strong>Explanation</strong>: The site default generation utility ended with warning conditions.</td>
<td><strong>System action</strong>: The site default generation utility ends with a return code of 4.</td>
<td><strong>User response</strong>: Check the preceding messages that explain the warning conditions. Correct the warning conditions and rerun the utility.</td>
<td></td>
</tr>
<tr>
<td>FABN1107E</td>
<td>\texttt{UNKNOWN KEYWORD IN SYSIN: keyword}</td>
<td>\textit{keyword}, which is specified in the SYSIN data set, is not a valid keyword.</td>
<td>The site default generation utility ends with a return code of 16.</td>
<td>Determine if the correct input control statement for site default is supplied. If necessary, correct the keyword and rerun the utility.</td>
</tr>
<tr>
<td></td>
<td><strong>Explanation</strong>: \textit{keyword}, which is specified in the SYSIN data set, is not a valid keyword.</td>
<td><strong>System action</strong>: The site default generation utility ends with a return code of 16.</td>
<td><strong>User response</strong>: Determine if the correct input control statement for site default is supplied. If necessary, correct the keyword and rerun the utility.</td>
<td></td>
</tr>
<tr>
<td>FABN109E</td>
<td>\texttt{UNKNOWN OPTION IN SYSIN: option}</td>
<td>\textit{option}, which is specified in the SYSIN data set, is not a valid option.</td>
<td>The site default generation utility ends with a return code of 16.</td>
<td>Determine if the correct input control statement for site default is supplied. If necessary, correct the option and rerun the utility.</td>
</tr>
<tr>
<td></td>
<td><strong>Explanation</strong>: \textit{option}, which is specified in the SYSIN data set, is not a valid option.</td>
<td><strong>System action</strong>: The site default generation utility ends with a return code of 16.</td>
<td><strong>User response</strong>: Determine if the correct input control statement for site default is supplied. If necessary, correct the option and rerun the utility.</td>
<td></td>
</tr>
<tr>
<td>FABN1030W</td>
<td>\texttt{INVALID RECORD IN SYSIN}</td>
<td>A record in an invalid format is in the SYSIN data set.</td>
<td>The site default generation utility skips this record and continues processing.</td>
<td>Determine if the correct input control statement for site default is supplied. If necessary, correct the record and rerun the utility.</td>
</tr>
<tr>
<td></td>
<td><strong>Explanation</strong>: A record in an invalid format is in the SYSIN data set.</td>
<td><strong>System action</strong>: The site default generation utility skips this record and continues processing.</td>
<td><strong>User response</strong>: Determine if the correct input control statement for site default is supplied. If necessary, correct the record and rerun the utility.</td>
<td></td>
</tr>
<tr>
<td>FABN1111E</td>
<td>\texttt{THE keyword STATEMENT IS INCORRECT}</td>
<td>Either the option that is specified for the indicated keyword is incorrect or the order of the keywords in the SYSIN data set is incorrect.</td>
<td>The site default generation utility ends with a return code of 16.</td>
<td>Ensure that the option specified for the indicated keyword is correct. If an OPTION statement is specified, ensure that the statement is located at the top of the SYSIN data set. Rerun the utility.</td>
</tr>
<tr>
<td></td>
<td><strong>Explanation</strong>: Either the option that is specified for the indicated keyword is incorrect or the order of the keywords in the SYSIN data set is incorrect.</td>
<td><strong>System action</strong>: The site default generation utility ends with a return code of 16.</td>
<td><strong>User response</strong>: Ensure that the option specified for the indicated keyword is correct. If an OPTION statement is specified, ensure that the statement is located at the top of the SYSIN data set. Rerun the utility.</td>
<td></td>
</tr>
<tr>
<td>FABN1200E</td>
<td>\texttt{FABNTGEN ENDED WITH ERRORS}</td>
<td>The site default generation utility ended with errors.</td>
<td>The site default generation utility ends with a return code of 8.</td>
<td>Check the preceding messages that explain the error conditions. Correct the error conditions and rerun the utility.</td>
</tr>
<tr>
<td></td>
<td><strong>Explanation</strong>: The site default generation utility ended with errors.</td>
<td><strong>System action</strong>: The site default generation utility ends with a return code of 8.</td>
<td><strong>User response</strong>: Check the preceding messages that explain the error conditions. Correct the error conditions and rerun the utility.</td>
<td></td>
</tr>
<tr>
<td>FABN1201E</td>
<td>\texttt{SITE DEFAULT TABLE FABNCTL0 IS CORRUPTED}</td>
<td>The site default table module (FABNCTL0) is corrupted and site default values are not reported.</td>
<td>The site default generation utility ends with a return code of 8.</td>
<td>Specify the correct FABNCTL0 module and ensure that the first 8 bytes of the site default table shows FABNCTL0. If the module is damaged, re-create another site default table module and store it in the STEPLIB data set.</td>
</tr>
<tr>
<td></td>
<td><strong>Explanation</strong>: The site default table module (FABNCTL0) is corrupted and site default values are not reported.</td>
<td><strong>System action</strong>: The site default generation utility ends with a return code of 8.</td>
<td><strong>User response</strong>: Specify the correct FABNCTL0 module and ensure that the first 8 bytes of the site default table shows FABNCTL0. If the module is damaged, re-create another site default table module and store it in the STEPLIB data set.</td>
<td></td>
</tr>
<tr>
<td>FABN1203E</td>
<td>\texttt{SITE DEFAULT TABLE FABNCTL0 IS NOT FOUND}</td>
<td>The site default values were not reported because the site default table module (FABNCTL0) is not in the STEPLIB data sets.</td>
<td>The site default generation utility ends with a return code of 8.</td>
<td>The site default generation utility ends with a return code of 8.</td>
</tr>
</tbody>
</table>
User response: Specify the data set that includes the site default table module (FABNCTL0) member to the STEPLIB statement.

FABN1204E  parameter_value IS NOT VALID FOR THE PARM PARAMETER OF THE EXEC STATEMENT

Explanation: An incorrect parameter is specified on the PARM= of the EXEC statement.

System action: The site default generation utility ends with a return code of 8.

User response: Specify either GEN or REPORT on the PARM parameter.

FABN1205E  NO VALID SITE DEFAULT CONTROL CARD IN SYSIN

Explanation: A control card that specifies the correct default option was not found in the SYSIN data set. The source code for the site default table is not created.

System action: The site default generation utility ends with a return code of 8.

User response: Specify correct control cards in the SYSIN data set.

FABN1250E  LOAD FAILED FOR DDNAME ddname MODULE module

Explanation: After a LOAD macro was issued to load the module module from the library that is specified by the ddname DD, register 15 contained a nonzero return code.

System action: The site default generation utility ends with a return code of 8.

User response: Ensure that the DD statement specifies the appropriate data set. Correct the error and rerun the job. If the problem persists, save the entire run listing (including the dump, JCL, and all reports from the site default generation utility), and contact IBM Software Support.

FABN3001E  UNABLE TO OPEN SYSIN DATA SET

Explanation: The SYSIN data set could not be opened during initialization.

System action: The site default generation utility ends abnormally.

User response: Determine the cause of the open failure. Correct the problem, and rerun the program.

FABN3002E  UNABLE TO OPEN SYSPUNCH DATA SET

Explanation: The SYSPUNCH data set could not be opened during initialization.

System action: The site default generation utility ends abnormally.

User response: Determine the cause of the open failure. Correct the problem, and rerun the program.

FABN3003E  UNABLE TO OPEN SYSOUT DATA SET

Explanation: The SYSOUT data set could not be opened during initialization.

System action: The site default generation utility ends abnormally.

User response: Determine the cause of the open failure. Correct the problem, and rerun the program.
# FABQ messages

Messages that are issued by the Advanced ACBGEN utility begin with the prefix FABQ.

<table>
<thead>
<tr>
<th>FABQ message</th>
<th>Explanation</th>
<th>System action</th>
<th>User response</th>
</tr>
</thead>
<tbody>
<tr>
<td>FABQ0001E UNABLE TO OPEN SYSPRINT -- JOB TERMINATED</td>
<td>Either the SYSPRINT DD statement is missing or OPEN processing failed for the data set that was specified.</td>
<td>Processing ends with an abend code of 949.</td>
<td>Ensure that the SYSPRINT DD exists, and that it specifies valid DD parameters. Correct any errors, and rerun the job.</td>
</tr>
<tr>
<td>FABQ0002E UNABLE TO OPEN DDNAME=ddname</td>
<td>The data set in the specified DD JCL statement could not be opened.</td>
<td>Processing ends with a return code of 16.</td>
<td>Ensure that the specified DD statement contains valid DD parameters. Correct any errors, and rerun the job.</td>
</tr>
<tr>
<td>FABQ0003E DDNAME=ddname NOT FOUND IN JCL</td>
<td>The specified DD statement could not be found in the JCL of the job step.</td>
<td>Processing ends with a return code of 16.</td>
<td>Ensure that the specified DD statement exists, and rerun the job.</td>
</tr>
<tr>
<td>FABQ0004W DDNAME=ddname DATASET EMPTY</td>
<td>This message is an informational message.</td>
<td>Processing continues, but the return code is set to 4.</td>
<td>None.</td>
</tr>
<tr>
<td>FABQ0005I OTHER REPORTS ARE PRINTED IN THE FABQRPT DATA SET BECAUSE GENDATE=YES</td>
<td>Because GENDATE=YES is specified in the LISTLIB command in the ACBSYSIN control statement, all of the reports except the Input Specifications report are in the FABQRPT data set.</td>
<td>Processing continues.</td>
<td>View other reports in the FABQRPT data set. If you did not specify the FABQRPT DD statement, the reports are printed to SYSPUT.</td>
</tr>
<tr>
<td>FABQ0006E DYNALLOC MACRO FAILED TO ALLOCATE ddname RC=rr RSN=ssss</td>
<td>The ddname data set was not dynamically allocated. rr is the hexadecimal return code, and ssss is the hexadecimal reason code of the DYNALLOC macro.</td>
<td>Processing ends with a return code of 16.</td>
<td>Find the reason for the dynamic allocation request failure. Use the information about the return codes and the reason codes in the z/OS MVS Programming: Authorized Assembler Services Guide to identify the cause of the error.</td>
</tr>
<tr>
<td>FABQ0007E DUMMY SPECIFIED FOR THE ddname DD STATEMENT</td>
<td>The reports are not generated because DUMMY is specified for the indicated DD statement.</td>
<td>Processing ends with a return code of 16.</td>
<td>Correct the DD statement so that it does not specify DUMMY, and then rerun the job.</td>
</tr>
<tr>
<td>FABQ0104E UNABLE TO LOAD NAME=module</td>
<td>An unsuccessful attempt was made to load the specified module. Standard MVS facilities were used to attempt the loading of the specified module.</td>
<td>Processing ends with a return code of 16.</td>
<td>Verify that the library containing the module is included in the STEPLIB DD JCL or in your MVS system’s link list.</td>
</tr>
<tr>
<td>FABQ0108E IMS ‘x.x.x’ NOT SUPPORTED</td>
<td>This version/release of IMS is not supported.</td>
<td>Processing ends with a return code of 16.</td>
<td>Ensure that your DFSRESLB DD statement contains the correct IMS load module library.</td>
</tr>
</tbody>
</table>
If the error persists, verify that the indicated version and release of IMS is supported by IMS Library Integrity Utilities. If the error persists, contact IBM Software Support.

**FABQ0109E IMS VERSION/RELEASE INCOMPATIBILITY**

*Explanation:* Several of the loaded IMS modules are from different versions and releases of IMS.

*System action:* Processing ends with a return code of 16.

*User response:* Ensure that the following IMS provided modules are all at the same IMS version/release level: DFSUACB0, DFSUAMB0, DFDSDLB0, and DFSVC000. If the error persists, contact IBM Software Support.

**FABQ0111E GETMAIN FAILED -- INSUFFICIENT MEMORY**

*Explanation:* A GETMAIN failed because of insufficient main memory.

*System action:* The job step ends with a user code of 16.

*User response:* Increase the MVS region size, and resubmit the job.

**FABQ0201F JOB TERMINATED DUE TO CONTROL STATEMENT ERRORS**

*Explanation:* This message is an informational message. The specific error is identified in a previous message.

*System action:* Processing ends with a return code of 16.

*User response:* Fix the problem reported in the previous FABQ0211W message, and rerun the job.

**FABQ0201W 'ddname' DOES NOT CONTAIN ANY COMMANDS**

*Explanation:* The specified DDName contained some data but no commands.

*System action:* Processing continues, but the return code is set to 4.

*User response:* Verify whether this condition is a valid condition. If not, correct the error, and rerun the job. If the error persists, contact IBM Software Support.

**FABQ0203E TOO MANY RECORDS IN THE CONTROL STATEMENT DATASET**

*Explanation:* The entire ACBSYSIN control data set is stored in memory while it is being processed. The anticipated maximum size was exceeded.

*System action:* Processing ends with a return code of 16.

*User response:* Verify that the correct data set is being used. If the error persists, contact IBM Software Support.

**FABQ0204E A CONTROL STATEMENT INDICATED CONTINUATION BUT NONE FOUND**

*Explanation:* The last record in the control data set indicated continuation; however, no continuation record was found.

*System action:* Processing ends with a return code of 16.

*User response:* See “Syntax rules” on page 384 for a description of the ACBSYSIN syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

**FABQ0205E INCOMPLETE COMMENT -- NO ENDING CHARACTERS FOUND**

*Explanation:* A control statement in the ACBSYSIN data set contained the beginning of a comment, but no comment-ending characters could be found.

*System action:* Processing ends with a return code of 16.

*User response:* See “Syntax rules” on page 384 for a description of the ACBSYSIN syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

**FABQ0210E NO COMMAND CODE PRESENT**

*Explanation:* A control statement in the ACBSYSIN data set contained a label field but did not contain a command code.

*System action:* Processing ends with a return code of 16.

*User response:* See “Syntax rules” on page 384 for a description of the ACBSYSIN syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

**FABQ0211E 'label' COMMAND LABEL IS INVALID**

*Explanation:* A control statement in the ACBSYSIN data set contains an incorrect label. The label field can contain no more than eight characters.

*System action:* Processing ends with a return code of 16.

*User response:* See “Syntax rules” on page 384 for a description of the ACBSYSIN syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.
FABQ0212E  *command* COMMAND CODE IS INVALID

Explanation: A control statement in the ACBSYSIN data set contains an unrecognizable command code.

System action: Processing ends with a return code of 16.

User response: See ["ACBSYSIN control statements" on page 384] for a description of the valid command codes. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

FABQ0220E  *keyword* KEYWORD IS INVALID

Explanation: A control statement in the ACBSYSIN data set contains an unrecognizable keyword operand.

System action: Processing ends with a return code of 16.

User response: See ["ACBSYSIN control statements" on page 384] for a description of the valid keyword operands. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

FABQ0221E  *keyword* KEYWORD DATA IS TOO LARGE

Explanation: A control statement in the ACBSYSIN data set contains a keyword operand whose data value is larger than allowed.

System action: Processing ends with a return code of 16.

User response: See ["ACBSYSIN control statements" on page 384] for a description of the valid keyword operand data values. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

FABQ0222E  *keyword* KEYWORD HAS MISSING RIGHT PAREN

Explanation: A control statement in the ACBSYSIN data set contains a keyword operand name whose data value is preceded by a left parenthesis, but there is no corresponding right parenthesis.

System action: Processing ends with a return code of 16.

User response: See ["Syntax rules" on page 384] for a description of the ACBSYSIN syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

FABQ0223E  *keyword* KEYWORD HAS MISSING RIGHT QUOTE

Explanation: A control statement in the ACBSYSIN data set contains a keyword operand name whose data value is preceded by a left quotation mark but no corresponding right quotation mark.

System action: Processing ends with a return code of 16.

User response: See ["Syntax rules" on page 384] for a description of the ACBSYSIN syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.
User response: See “Syntax rules” on page 384 for a description of the ACBSYSIN syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ0231E 'keyword' KEYWORD IS MUTUALLY EXCLUSIVE WITH A PREVIOUS KEYWORD

Explanation: A control statement in the ACBSYSIN data set contains two keyword operands that are mutually exclusive.

System action: Processing ends with a return code of 16.

User response: See “ACBSYSIN control statements” on page 384 for a description of the syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ0232E 'keyword' KEYWORD SPECIFIED MORE THAN ONCE

Explanation: A control statement in the ACBSYSIN data set contains the same keyword operand more than once.

System action: Processing ends with a return code of 16.

User response: Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ0233E 'keyword' KEYWORD HAS TOO MANY PARAMETERS

Explanation: A control statement in the ACBSYSIN data set contains a keyword operand that has too many data values.

System action: Processing ends with a return code of 16.

User response: See “ACBSYSIN control statements” on page 384 for a description of the syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ0234E 'keyword' KEYWORD HAS INVALID DATA

Explanation: A control statement in the ACBSYSIN data set contains a keyword operand that has incorrect data.

System action: Processing ends with a return code of 16.

User response: See “ACBSYSIN control statements” on page 384 for a description of the syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ0235E 'keyword' KEYWORD HAS INVALID DATA

Explanation: A control statement in the ACBSYSIN data set contains a keyword operand that has incorrect data.

System action: Processing ends with a return code of 16.

User response: See “ACBSYSIN control statements” on page 384 for a description of the syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ0300E SORT PROGRAM TERMINATED WITH RC=xx

Explanation: The sort program that was called by FABQCHEK returned an RC=xx, where xx is the return code of the sort program.

System action: Processing continues, but the nonzero sort return code will be the condition code of the job step.

User response: Check the condition and determine whether if any action must be taken.

FABQ1001I ABOVE ERROR OCCURRED IN RCD #nnn, AT OR NEAR POSITION nn

Explanation: This message is an informational message that describes where the error occurred for the preceding message.

System action: Processing continues.

User response: Fix the problem reported in the preceding FABQnnnn message, and rerun the job.

FABQ1002E LABEL NAME TOO LONG

Explanation: The label field in a SYSIN control statement contains more than eight characters. The label field, if present, must consist of a period (.) followed by 1 - 7 alphanumeric characters.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ1003E INVALID LABEL NAME

Explanation: The label field in a SYSIN control statement contained an incorrect character. The label field, if present, must consist of a period (.) followed by 1 - 7 alphanumeric characters.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ1003E NO OPERATION NAME FOUND

Explanation: No operation field was found in a SYSIN control statement.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.
ABQ1004E  INVALID OPERATION NAME

Explanation: The operation field in a SYSIN control statement contains an unrecognizable command.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

ABQ1005E  EXPECTED OPERAND NOT FOUND

Explanation: An operand in a SYSIN control statement was terminated with a continuation character but no further operands were found in the statement.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

ABQ1006E  OPERAND NAME TOO LONG

Explanation: An operand name in a SYSIN control statement contains more than eight characters.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

ABQ1007E  OPERAND NAME NOT TERMINATED WITH AN "=" CHAR

Explanation: An operand name in a SYSIN control statement was not immediately followed by an equal sign (=).

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

ABQ1008E  INVALID OPERAND NAME

Explanation: An operand name in a SYSIN control statement is unrecognizable.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

ABQ1009E  OPERAND HAS NO DATA

Explanation: An operand name in a SYSIN control statement has no data.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

ABQ1010E  INVALID OPERAND DATA

Explanation: The data in an operand in a SYSIN control statement contains incorrect data.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

ABQ1011E  OPERAND HAS MISSING RIGHT PAREN

Explanation: The data in an operand in a SYSIN control statement is not enclosed in a balanced pair of parentheses.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

ABQ1012E  MULTIPLE OPERAND VALUES MUST BE ENCLOSED IN PARENS

Explanation: If there is more than one value present, the data in an operand in a SYSIN control statement must be enclosed in a balanced pair of parentheses.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

ABQ1013E  MORE OPERAND DATA EXPECTED -- NONE FOUND

Explanation: A data value in an operand in a SYSIN control statement was terminated with a continuation character; however, no more data was found.

System action: Processing continues with the next control statement.

User response: For a description of the syntax rules, see IMS System Utilities. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.
FABQ1014E 1ST 15 POSITIONS OF CONTINUATION MUST BE BLANK

**Explanation:** A continuation record in a SYSIN control statement must be blank in the first 15 positions.

**System action:** Processing continues with the next control statement.

**User response:** For a description of the syntax rules, see *IMS System Utilities*. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ1015E POSITION 16 OF CONTINUATION MUST NOT BE BLANK

**Explanation:** A continuation record in a SYSIN control statement must start in position 16.

**System action:** Processing continues with the next control statement.

**User response:** For a description of the syntax rules, see *IMS System Utilities*. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ1016E INVALID DELIMITING CHARACTER

**Explanation:** An operand name in a SYSIN control statement must be followed by an equal sign (=), and the operand data must be terminated with a blank or a comma.

**System action:** Processing continues with the next control statement.

**User response:** For a description of the syntax rules, see *IMS System Utilities*. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABQ1025W DBD/PSB ‘member’ REFERENCED MORE THAN ONCE

**Explanation:** The PSB or DBD name appears more than once in either BUILD statements or DELETE statements. The same name can appear in both a BUILD and DELETE statement, but not more than once per BUILD or DELETE list.

**System action:** The second and later occurrences are discarded, and processing continues, but generates a return code of 4.

**User response:** None.

FABQ1030W DMB ‘dmbname’ IN ACBLIB IS NOT REFERENCED BY ANY PSB

**Explanation:** The specified DMB was found in the ACB library but was not referred to by any PSBs found in the same ACB library.

**System action:** Processing continues.

**User response:** Check the condition, and determine whether any action must be taken.

FABQ1031W DMB ‘dmbname’ REFERENCED BY PSB ‘psbname’ IS NOT IN ACBLIB

**Explanation:** The specified PSB referred to the specified DMB, but the DMB was not found in the ACB library.

**System action:** Processing continues, but the return code is set to 8.

**User response:** Check the condition and determine if any action must be taken.

FABQ1032W PSB ‘psbname’ REFERS TO ‘dmbname’ BUT IT IS NOT A DMB

**Explanation:** The specified PSB refers to the specified DMB, but the directory entry for the DMB shows that it is not a DMB.

**System action:** Processing continues, but the return code is set to 8.

**User response:** Check the condition and determine whether any action must be taken.

FABQ1033W DIRECTORY ENTRY ‘mbrname’ HAS DIFFERENT IMS LEVEL

**Explanation:** The IMS release and level found in the first ACB library directory entry is used to compare against all other directory entries. The specified entry differs from the first entry.

**System action:** Processing continues, but the return code is set to 4.

**User response:** Check the condition and determine if any action must be taken.

FABQ1034W DIRECTORY ENTRY ‘mbrname’ HAS A BAD SIZE

**Explanation:** All directory entries in the library specified in the INDD= operand are checked for valid size. The directory entry size for a DBD/PSB library is different from the directory entry size for the ACB library.

**System action:** Processing continues, but the return code is set to 8. The member identified in the message is skipped and therefore is not reflected in the reports.

**User response:** Verify that the INDD= and LIBTYPE= operands in the LISTLIB command are consistent. For example, if LIBTYPE=ACB was specified, verify that the INDD= operand specifies the ddname of an ACB library.
**FABQ1036W** HI-KEY DIRECTORY ENTRY NOT LAST IN DIRECTORY RECORD

**Explanation:** An incorrect directory block was encountered. An X’FF’ directory entry was encountered, but the length field in the directory block showed that there were more directory entries.

**System action:** Processing continues, but the return code is set to 8.

**User response:** Check the condition and determine whether any action must be taken.

**FABQ1037W** EMPTY DIRECTORY RECORD ENCOUNTERED BEFORE HI-KEY ENTRY

**Explanation:** An incorrect directory block was encountered. Empty directory blocks can appear only after the block containing the X’FF’ hi-key entry.

**System action:** Processing continues, but the return code is set to 8.

**User response:** Check the condition and determine whether any action must be taken.

**FABQ1038W** NO DIRECTORY RECORDS FOUND

**Explanation:** The input library is probably not a valid partitioned data set, because directory records (blocks) are created when the PDS is allocated.

**System action:** Processing continues, but the return code is set to 8.

**User response:** Check the condition and determine whether any action must be taken.

**FABQ1039W** NO HI-KEY DIRECTORY ENTRY FOUND

**Explanation:** The input library did not contain an X’FF’ hi-key directory entry.

**System action:** Processing continues, but the return code is set to 8.

**User response:** Check the condition and determine whether any action must be taken.

**FABQ1040W** INVALID DIRECTORY KEY FOUND

**Explanation:** The input library contains a directory entry whose first byte in the key contains X’FF’, but the rest is not X’FF’.

**System action:** Processing continues, but the return code is set to 8.

**User response:** Check the condition and determine whether any action must be taken.

**FABQ1041E** DD CONCATENATION NOT SUPPORTED -- DDNAME=ddname

**Explanation:** Concatenation is not supported for the specified DD statement.

**System action:** Processing ends with a return code of 16.

**User response:** Correct the JCL, and rerun the job.

**FABQ1042W** ACBGEN SUBTASK UNABLE TO LOAD ITS OWN COPY OF DFSDLBL0

**Explanation:** Concatenation of the input library being analyzed is not supported.

**System action:** Processing continues, but the second and subsequent DD statements in the concatenation are ignored.

**User response:** If you want to analyze multiple libraries, it must be done one per job step.

**FABQ9900E** SYSPRINT PERMANENT I/O ERROR

**Explanation:** The SYSPRINT data set had a permanent I/O error.

**System action:** ACBGEN ends abnormally with a user code of 0948.

**User response:** Check LOGREC to determine the cause of the error, and resubmit the job.

**FABQ9901E** DFSPRINT PERMANENT I/O ERROR

**Explanation:** The DFSPRINT data set had a permanent I/O error.

**System action:** ACBGEN ends abnormally with a user code of 0948.

**User response:** Check LOGREC to determine the cause of the error, and resubmit the job.

**FABQ9908E** INVALID DFS-TYPE MESSAGE ENCOUNTERED

**Explanation:** A message ID was not found in the DFSUMGT0 message text module.

**System action:** ACBGEN ends abnormally with a user code of 0944.

**User response:** Resubmit the job, using the IMS ACBGEN utility. If you get an IMS 0944 abend, the problem is in the DFSUMGT0 module. If the IMS ACBGEN utility does not abend, there is a logic error in IMS Library Integrity Utilities and you must contact IBM Software Support.
**FABQ9909E** TOO FEW UERR VARIABLES

**Explanation:** An incorrect number of parameters were passed to the message formatter module by the UERR macro.

**System action:** ACBGEN ends abnormally with a user code of 0945.

**User response:** Resubmit the job, using the IMS ACBGEN utility. If you get an IMS 0945 abend, the problem is in an IMS module. If the IMS ACBGEN utility does not abend, there is a logic error in IMS Library Integrity Utilities. Contact IBM Software Support.

---

**FABQ9910E** FABQDRIV LOGIC ERROR

**Explanation:** An internal logic error.

**System action:** ACBGEN ends abnormally with a user code of 1000.

**User response:** Contact IBM Software Support.

---

**FABQ9911E** UNABLE TO HOOK DFSUACB0

**Explanation:** FABQHUK1 was unable to hook the DFSUACB0 load module.

**System action:** ACBGEN ends abnormally with a user code of 1001 through 1007.

**User response:** Verify that the DFSRESLIB DD statement is specified and that you are using the appropriate IMS RESLIB. If the problem persists, add a 'DBTSNAP DD SYSOUT=a' DD statement, and resubmit the job; then contact IBM Software Support.

---

**FABQ9912E** UNABLE TO HOOK DFSDLBL0

**Explanation:** FABQHUK2 was unable to hook the DFSDLBL0 load module.

**System action:** ACBGEN ends abnormally with a user code of 1011 through 1016.

**User response:** Verify that you are using the appropriate IMS RESLIB. If the problem persists, add a 'DBTSNAP DD SYSOUT=a' DD statement, and resubmit the job; then contact IBM Software Support.

---

**FABQ9913W** NO DMB FOUND FOR DFS0960 MESSAGE

**Explanation:** An internal logic error occurred in FABQRPT2.

**System action:** Processing continues.

**User response:** Check your output thoroughly. Contact IBM Software Support and report this message.

---

**FABQ9914W** ACBGEN DIRECTORY SYNC ERROR#1

**Explanation:** An internal logic error occurred in FABQTASK.

**System action:** Processing continues.

**User response:** Check your output thoroughly. Contact IBM Software Support and report this message.

---

**FABQ9915W** ACBGEN DIRECTORY SYNC ERROR#2

**Explanation:** An internal logic error occurred in FABQTASK.

**System action:** Processing continues.

**User response:** Check your output thoroughly. Contact IBM Software Support and report this message.

---

**FABQ9916W** ACBGEN DIRECTORY SYNC ERROR#3

**Explanation:** An internal logic error occurred in FABQDRIV.

**System action:** Processing continues.

**User response:** Check your output thoroughly. Contact IBM Software Support and report this message.

---

**FABQ9917E** RDJFCB FAILED FOR DDNAME ddname

**Explanation:** The READJFCB failed for a DDNAME ddname data set.

**System action:** ACBGEN ends abnormally with a system code of 0C3.

**User response:** Contact IBM Software Support.

---

**FABQ9920E** EOF ENCELLED READING PSB DIRECTORY BLOCKS

**Explanation:** An internal logic error occurred in FABQPLST.

**System action:** ACBGEN ends abnormally with a system code of 0C3.

**User response:** Contact IBM Software Support.

---

**FABQ9921E** EOF ENCELLED READING ACB DIRECTORY BLOCKS

**Explanation:** An internal logic error occurred in FABQPLST.

**System action:** ACBGEN ends abnormally with a system code of 0C3.

**User response:** Contact IBM Software Support.
FABQ9922E  ACBGEN STOW ERROR IN-MEMORY DIRECTORY
Explanation: An internal logic error occurred in FABQPLST.
System action: ACBGEN ends abnormally with a system code of 0C3.
User response: Contact IBM Software Support.

FABQ9923E  ACBGEN BLDL ERROR
Explanation: An internal logic error occurred in FABQPLST.
System action: ACBGEN ends abnormally with a system code of 0C3.
User response: Contact IBM Software Support.

FABQ9924E  ACBGEN BLDL ERROR
Explanation: An internal logic error occurred in FABQPLST.
System action: ACBGEN ends abnormally with a system code of 0C3.
User response: Contact IBM Software Support.

FABQ9925E  ACBGEN STOW ERROR
Explanation: An internal logic error occurred in FABQPLST.
System action: ACBGEN ends abnormally with a system code of 0C3.
User response: Contact IBM Software Support.

FABQ9926E  ACBGEN ENCLOSED EMPTY DIRECTORY BLOCK
Explanation: An internal logic error occurred in FABQPLST.
System action: ACBGEN ends abnormally with a system code of 0C3.
User response: Contact IBM Software Support.

FABQ9927E  ACBGEN PDS READ ERROR
Explanation: An internal logic error occurred in FABQPLST.
System action: ACBGEN ends abnormally with a system code of 0C3.
User response: Contact IBM Software Support.

FABQ9928E  ACBGEN FOUND 2 SEQ LIST TTRS WITH SAME VALUE
Explanation: An internal logic error occurred in FABQPLST.
System action: ACBGEN ends abnormally with a system code of 0C3.
User response: Contact IBM Software Support.

FABQ9929E  ACBGEN UNABLE TO INSERT INTO DELETE LIST
Explanation: An internal logic error occurred in FABQPLST.
System action: ACBGEN ends abnormally with a system code of 0C3.
User response: Contact IBM Software Support.

FABQ9937E  PUT FAILED FOR DDNAME: ddname
Explanation: The PUT macro failed for the indicated data set.
System action: ACBGEN ends abnormally with a user code of 1027.
User response: Check the status of the indicated data set, and by referring to the MVS system message and its programmer response, correct the error. Rerun the job.

FABQ9991W  An ACBGEN SUBTASK ABENDED WITH SYSTEM CODE OF "xxx" FOR PSB=psbname
Explanation: This message is issued whenever a subtask aborts. The abend code is displayed in the message. The PSB it was processing is also shown in the message.
System action: The subtask which issued this MVS console message ends but processing continues in the other subtasks.
User response: Correct the problem that caused the abend, and either rerun the job or resubmit an
ACBGEN job to generate the PSB that failed.

---

**FABQ9992I** NEW ACB LIBRARY DIRECTORY INCOMPLETE

**Explanation:** This message is issued during termination processing. The reporting modules encountered an incomplete directory in the ACB library. The typical cause for this condition is an ACB library that did not have sufficient space to hold all DMBs, PSBs, or both.

**System action:** The reporting process continues. Any previously generated return codes are honored.

**User response:** Check the SYSPRINT data set for any DFS-type messages that suggest what might have caused this condition.

---

**FABQ9993I** BUILD PROCESS FOR PSB psbname IS RESTARTING

**Explanation:** This message indicates that Advanced ACBGEN resolved the storage shortage problem (notified by a DFS0649W message) and the build process is restarted for the indicated PSB.

**System action:** The build PSB process restarts.

**User response:** None. This message is informational.

---

**FABQ9997I** mmmm OF nnnn PSBS PROCESSED BY ACBGEN

**Explanation:** This message is issued when the MONITOR=(PROGRESS=(Y,frequency_value)) operand is specified in the ACBSYSIN data set. mmmm is the number of the members that were specified either explicitly or implicitly in the SYSIN data set and that will be processed by the ACBGEN utility.

If BUILD DBD=dbname,BLDPSB=NO is specified and if PSBs reference the DBDs for Fast Path DEDBs or shared secondary index databases that do not change the physical structure of database, the ACBGEN utility skips rebuilding such PSBs during the PSB process. In this case, the number of members that are processed will be fewer than nnnn.

**System action:** This message is issued to the MVS console when the ACBGEN process begins.

**User response:** None. This message is informational.

---

**FABQ9998I** mmmm OF nnnn PSBS PROCESSED BY ACBGEN

**Explanation:** This message is issued when the MONITOR=(PROGRESS=(Y,frequency_value)) operand is specified in the ACBSYSIN data set. mmmm is the number of the members that were specified either explicitly or implicitly in the SYSIN data set and that will be processed by the ACBGEN utility.

If BUILD DBD=dbname,BLDPSB=NO is specified and if PSBs reference the DBDs for Fast Path DEDBs or shared secondary index databases that do not change the physical structure of database, the ACBGEN utility skips rebuilding such PSBs during the PSB process. In this case, the number of members that are processed will be fewer than nnnn.

**System action:** This message is issued to the MVS console when the ACBGEN process ends.

**User response:** None. This message is informational.
FABV messages

Messages that are issued by the MFS Reversal utility and the MFS Compare utility begin with the prefix FABV.

FABV0000E  INVALID ERROR MESSAGE NUMBER PASSED

Explanation: An internal error has occurred.

System action: Processing continues with the current control block. The name of the control block is indicated in a subsequent message FABV0044I. The source generated for this control block might be in error.

User response: Contact IBM Software Support for assistance in determining whether the source was or was not built for the current control block.

Module: FABVLOG

FABV0001E  OPEN FAILED FOR DDNAME=ddname.

Explanation: The specified data set could not be opened.

System action: Processing terminates.

User response: Check that the data set defined by the DD name ddname is allocated and that the DSORG of the data set conforms to the program requirements.

Module: FABVCNTL

FABV0004E  BLDL FAILED FOR MBR=member.

Explanation: The indicated member was not found in the format library.

System action: The utility is terminated.

User response: Try running the utility again. If the problem persists, contact IBM Software Support for assistance.

Module: FABVBLDC, FABVCOMP

FABV0005E  FIND FAILED FOR MBR=member.

Explanation: The indicated member was not found in the format library.

System action: The utility is terminated.

User response: Try running the utility again. If the problem persists, contact IBM Software Support.

Module: FABVBLDC, FABVCOMP

FABV0006E  READ ERROR WHILE READING MBR=member.

Explanation: A READ error occurred while reading the indicated member. This error could be due to a number of reasons.

System action: The utility terminates.

User response: Try running the utility again. If the problem persists, contact IBM Software Support for assistance.

Module: FABVBLDC, FABVCOMP

FABV0007E  WRITE ERROR TO DD=MFSSRCE

R0=reasoncode  R15=returncode

Explanation: A WRITE error occurred while writing to data set MFSSRCE. The return code and reason codes are indicated.

System action: The utility terminates.

User response: Check the return and reason codes in z/OS DFSMS Macro Instructions for Data Sets to identify the cause of the failure.

Module: FABVBLDS

FABV0008E  STOW FAILED FOR MBR=member.

R0=reasoncode  R15=returncode

Explanation: A STOW error occurred for the specified member. The return code and reason codes are indicated.

System action: The utility terminates.

User response: Check the return and reason codes in z/OS DFSMS Macro Instructions for Data Sets to identify the cause of the failure.

Module: FABVBLDC

FABV0009E  DIF/DOF DATA ERROR, BYPASSED MBR=member

Explanation: This member is unrecognizable by the MFS Reversal utility. The member was bypassed.

System action: The utility continues with the next format library member. When the MFS Reversal utility is run without the OPTION NOXRPT statement, the utility stops to analyze and report the cross-reference information between the MIDs and MODs that are referenced by the member and continues with the next format.

User response: None.

Module: FABVBLDC, FABVDIRC, FABV3270

FABV0010E  READ ERROR FROM DD=SYSIN

R0=reasoncode,  R15=returncode

Explanation: A READ error occurred for the specified
member. The return code and reason codes are indicated.

System action: The utility terminates.

User response: Check the return and reason codes in z/OS DFSMS Macro Instructions for Data Sets to identify the cause of the failure.

Module: FABVANL

FABV0011E INVALID DEVICE TYPE devicetype FOR MEMBER member

Explanation: An invalid device type was encountered.

System action: The source build for the current DIF/DOF cannot proceed, and so it is terminated. The utility continues processing with the next DIF/DOF. When the MFS Reversal utility is run without the OPTION NOXRPT statement, the utility stops to analyze and report the cross-reference information between the MIDs and MODs that are referenced by the member and continues with the next format.

User response: The cross-reference list written to SYSPRINT indicates the device with an invalid (unknown) device type. Either delete the format so that this error does not occur in the future, or ignore this message.

Module: FABVFIDO, FABVFINI, FABVBLDC, FABVDMAI, FABVDMAO, FABVDMBI, FABVDMBO, FABVFIO

FABV0016W DUPLICATE MEMBER member SPECIFIED ON SELECT.

Explanation: A SELECT statement was processed, but it contained duplicate member names.

System action: The second appearance of the name is ignored and processing continues.

User response: None.

Module: FABVANL

FABV0017W MEMBER member SPECIFIED ON SELECT IS NOT IN DIRECTORY.

Explanation: A SELECT statement was processed which contained a member that is not found in the directory of the format library.

System action: The indicated member is ignored and processing continues.

User response: Check the specification of the member name to make sure that it is correct.

Module: FABVANL

FABV0018E INVALID FEATURE CODE feature.

Explanation: An invalid feature was detected in the specification.

System action: The source build for the current DIF/DOF cannot proceed, and so it is terminated. The utility continues processing with the next DIF/DOF. When the MFS Reversal utility is run without the OPTION NOXRPT statement, the utility stops to analyze and report the cross-reference information between the MIDs and MODs that are referenced by the member and continues with the next format.

User response: The cross-reference list written to SYSPRINT indicates the device with an invalid (unknown) feature. Either delete the format so that this error does not occur in the future, or ignore this message.

Module: FABVFIDO, FABVFINI, FABVBLDC, FABVDMAI, FABVDMAO, FABVDMBI, FABVDMBO, FABVFIO

FABV0020W INVALID KEYWORD keyword IN UTILITY CONTROL STATEMENT.

Explanation: An invalid keyword was specified on the control statement.

System action: The control statement containing the invalid keyword is ignored and processing continues with the next control statement.

User response: Correct the specification of the keyword and rerun the utility.

Module: FABVANL

FABV0021E CANNOT RESOLVE BUFFER ADDRESS: MSG=member FMT=format DEV=device

Explanation: The number of lines for the display device for which the source is being built is 0. The buffer address of a field could not be resolved.

System action: The source build for the current DIF/DOF cannot proceed, and so it is terminated. The utility continues processing with the next DIF/DOF.

User response: The cross-reference list written to SYSPRINT lists the devices referenced by the selected MID/MOD. Check if a device of the type 3270-An is reported. Ensure that the Device Characteristics Table DFSUDT0x or the default table FABVDFCT has all the definitions of 3270-Ax listed. Try running the utility again with the correct Device Characteristics Table.

Module: FABVTRBB
FABV0022W USING DEFAULT DEVICE CHARACTERISTICS TABLE
FABV0DVC

Explanation: The module DFSUDT0x could not be loaded, x having either the default value of A or the value specified on the DVCTBL utility input statement.

System action: Processing continues using the default Device Characteristics Table.

User response: If your system does not have 3270-An type devices, this message is of no significance. If your system does have 3270-An type devices, verify the existence of the DFSUDT0x module and specify the current suffix on the DVCTBL utility input statement.

Module: FABVCNL

FABV0030W MID/DIF TIMESTAMP NOT THE SAME FOR MBR member

Explanation: The time stamp that indicates when the MID was generated differs from the time stamp of the DIF that is referenced by this MID. member shows the name of the MID.

System action: The utility does not build the source for the format that is referenced by the MID and the message descriptors that are associated with the format. The utility continues processing. When the MFS Reversal utility is run without the OPTION NOXRPT statement, the utility stops to analyze and report the cross-reference information of the MID and continues with the next format.

User response: The difference between the time stamps indicates that the MFS control blocks might be inconsistent. Consider regenerating the format that is referenced by the MID and the message descriptors that are associated with the format.

Module: FABVCNL

FABV0031W MOD/DOF TIMESTAMP NOT THE SAME FOR MBR member

Explanation: The time stamp that indicates when the MOD was generated differs from the time stamp of the DOF that is referenced by this MOD. member shows the name of the MOD.

System action: The utility does not build the source for the format that is referenced by the MOD and the message descriptors that are associated with the format. The utility continues processing. When the MFS Reversal utility is run without the OPTION NOXRPT statement, the utility stops to analyze and report the cross-reference information of the MOD and continues with the next format.

User response: The difference between the time stamps indicates that the MFS control blocks might be inconsistent. Consider regenerating the format that is referenced by the MOD and the message descriptors that are associated with the format.

Module: FABVCNL

FABV004I SOURCE FOR MEMBER member BUILT.

Explanation: MFS Reversal has successfully generated source code for the specified member. This message is an informational message.

System action: The utility continues with the source build for the next MID/MOD specified on the SELECT statement.

User response: None. This message is informational.

Module: FABVCNL

FABV004E NON-ZERO RETURN CODE FROM DYNALLOC.

Explanation: The MVS macro DYNALLOC, which is used for determining file characteristics, returned a non-zero return code.

System action: Processing terminates.

User response: Try running the utility again. If the problem persists, contact IBM Software Support for assistance.

Module: FABVQRY

FABV004E NO SELECTED MEMBER FOUND IN FORMAT LIBRARY.

Explanation: None of the selected members was found in the format library.

System action: Processing terminates.

User response: Check the names of the MID and MODs on the SELECT statement. Also check that the format library is correctly specified.

Module: FABVANL

FABV004E EMPTY SELECT STATEMENT

Explanation: No member name is specified in the SELECT statement.

System action: Processing terminates.

User response: None.

Module: FABVANL

FABV0054E FORMAT LIB HAS NO MEMBERS

Explanation: The library specified with FORMAT DD contains no member.

System action: Processing terminates.

User response: None.

Module: FABVCNTL
FABV0060I COPY PROCESS STARTED
Explanation: The copy process started.
System action: The copy process continues.
User response: None. This message is informational.
Module: FABVCOPY

FABV0061I COPY PROCESS COMPLETED
Explanation: The copy process completed.
System action: The copy process ends and the utility continues processing.
User response: None. This message is informational.
Module: FABVCOPY

FABV0062W COPY PROCESS COMPLETED, BUT SOME WARNINGS WERE DETECTED
Explanation: The copy process completed with warning conditions.
System action: The copy process ends and the utility continues processing.
User response: Check the preceding warning messages. If necessary, correct the warning conditions and rerun the utility.
Module: FABVCOPY

FABV0063E COPY PROCESS COMPLETED, BUT SOME ERRORS WERE DETECTED
Explanation: The copy process completed, but one or more errors were detected.
System action: The copy process ends and the utility continues processing.
User response: Check the preceding error messages. If necessary, correct the error conditions and rerun the utility.
Tip: Incomplete members might be created in the partitioned data set that is specified by the COPYFMT DD statement. The presence of these members might lead to poor performance or shortage of space when rerunning the copy function. If performance degradation or space shortage is a concern, remove the existing members and redefine the data set before you rerun the job.
Module: FABVCOPY

FABV0064E COPY PROCESS ENDED BECAUSE ERRORS WERE DETECTED
Explanation: The copy process ended abnormally with one or more errors. The COPYPRT DD statement was specified for the job, but a copy report is not generated.
System action: The copy process ends abnormally. The utility does not generate a copy report.
User response: Check the preceding error messages. Correct the error conditions and rerun the utility.
Tip: Incomplete members might be created in the partitioned data set that is specified by the COPYFMT DD statement. The presence of these members might lead to poor performance or shortage of space when rerunning the copy function. If performance degradation or space shortage is a concern, remove the existing members and redefine the data set before you rerun the job.
Module: FABVCOPY

FABV0065E COPY PROCESS ENDED BECAUSE COPYFMT LIBRARY CONTAINED SOME ALIAS MEMBERS
Explanation: The copy process ended abnormally because some alias members, which are not supported by the copy function, were detected in the partitioned data set that is specified in the COPYFMT DD statement. No MFS control blocks are copied to the partitioned data set. The COPYPRT DD statement was specified for the job, but a copy report is not generated.
System action: The copy process ends abnormally.
User response: Check the partitioned data set that is specified in the COPYFMT DD statement. If necessary, delete the alias members and rerun the utility.
Module: FABVCOPY

FABV0070I COPY COMPLETED FOR MBR=member
Explanation: The indicated member was copied to the partitioned data set that is specified by the COPYFMT DD statement.
System action: The utility continues processing.
User response: None. This message is informational.
Module: FABVCOPY

FABV0071I SUCCESSFULLY REPLACED MBR=member
Explanation: The indicated member, which resides in the partitioned data set that is specified by the COPYFMT DD statement, is replaced.
System action: The utility continues processing.
User response: None. This message is informational.
Module: FABVCOPY
COPY SKIPPED FOR MBR=member, THE MEMBER ALREADY EXISTS

Explanation: The indicated member exists in the partitioned data set that is specified by the COPYFMT DD statement. Because the REPLACE=YES option is not specified in the SYSIN DD, the copy process skipped this member.

System action: The utility continues processing.

User response: None. This message is informational.

Module: FABVCOPY

COPY SKIPPED FOR MBR=member, THE MEMBER TYPE IS ALIAS

Explanation: The indicated member, which resides in the format library that is specified by the FORMAT DD statement, is an alias member. The copy process skipped this member because the copy function does not support alias members.

System action: The utility continues processing.

User response: If you want to copy the alias member, re-create the alias member manually.

Module: FABVCOPY

MBR=member1 THAT IS REFERENCED BY MBR=member2 DOES NOT EXIST IN DD=FORMAT

Explanation: Member member1 that is referenced by member member2 does not exist in the format library that is specified by the FORMAT DD statement.

System action: The utility continues processing.

User response: Check the format library and, if necessary, re-create the member.

Module: FABVCOPY

BLDL ERROR WHILE COPYING
MBR=member IN DD=[FORMAT | COPYFMT] RSN=reason_code RC=return_code

Explanation: An error occurred while the BLDL macro was retrieving the directory information about member member that resides in the format library that is specified by the FORMAT DD statement or the partitioned data set that is specified by the COPYFMT DD statement. reason_code shows the reason code, and return_code shows the return code from the macro.

System action: The utility continues with the next format library member.

User response: See the topic "BLDL completion codes" in z/OS DFSMS Macro Instructions for Data Sets to determine the cause of the BLDL macro failure. Correct the error, and rerun the utility. If the problem persists, contact IBM Software Support.

Module: FABVCOPY

READ ERROR WHILE COPYING
MBR=member IN DD=FORMAT

Explanation: An error occurred while the READ macro was reading member member that resides in the format library that is specified by the FORMAT DD statement.

System action: The utility continues with the next format library member.

User response: Determine the cause of the READ macro failure, correct the error, and rerun the utility. If the problem persists, contact IBM Software Support.

Module: FABVCOPY

WRITE ERROR WHILE COPYING
MBR=member TO DD=COPYFMT

Explanation: An error occurred while a WRITE macro was copying member member to the partitioned data set that is specified by the COPYFMT DD statement.

System action: The utility ends abnormally.

User response: Determine the cause of the WRITE macro failure, correct the error, and rerun the utility. If the problem persists, contact IBM Software Support.

Module: FABVCOPY

STOW ERROR WHILE COPYING
MBR=member TO DD=COPYFMT RSN=reason_code RC=return_code ID=id

Explanation: An error occurred in the STOW macro that was issued while the copy process was copying member member to the partitioned data set that is specified in the COPYFMT DD statement. reason_code shows the reason code, and return_code shows the return code from the macro. id is an identifier that is associated with the internal location where the STOW macro was issued.

System action: The utility ends abnormally.

User response: See the topic "STOW completion codes" in z/OS DFSMS Macro Instructions for Data Sets to determine the cause of the STOW macro failure. Correct the error, and rerun the utility. If the problem persists, contact IBM Software Support.

Module: FABVCOPY

READ ERROR WHILE COPYING IN DD=COPYFMT

Explanation: An error occurred while the READ macro was reading the directory entry of the
partitioned data set that is specified by COPYFMT DD statement.

**System action:** The utility ends abnormally.

**User response:** Determine the cause of the READ macro failure, correct the error, and rerun the utility. If the problem persists, contact IBM Software Support.

**Module:** FABVCOPY

---

**FABV0099E  INVALID PROGRAM INVOCATION**

**Explanation:** An attempt was made to invoke the Reversal or Compare utility incorrectly.

**System action:** Processing terminates.

**User response:** Correct the JCL used to invoke the utility and try running the utility again.

**Module:** FABVCNTL

---

**FABV3000E  GETMAIN FAILED**

**Explanation:** The program could not obtain sufficient area with the GETMAIN macro.

**System action:** The MFS Reversal utility ends abnormally.

**User response:** Increase the region size on the JOB or the EXEC statement in the JCL, and rerun the utility.

**Module:** FABVCNTL, FABVMREF

---

**FABV9000I  EXCLUDED MEMBER=**

**Explanation:** The utility skips processing the member member of a MID, MOD, DIF, or DOF that is specified in the EXCLUDE statement.

**System action:** Processing continues.

**User response:** None. This message is informational.

**Module:** FABVBLDC
FABW messages
Messages that are issued by the Multiple Resource Checker utility begin with the prefix FABW.

FABW0001I  CONTROL STATEMENT SUPPLIED IS: control statement
Explanation: This message is the echo of the FABWCTL control statements that are processed by the Multiple Resource Checker utility.
System action: The Multiple Resource Checker utility continues processing.
User response: None. This message is informational.

FABW0002I  PARAMETER USED IS: parameter
Explanation: The Multiple Resource Checker utility proceeds with the indicated parameter.
System action: The Multiple Resource Checker utility continues processing.
User response: None. This message is informational.

FABW0003I  NO DIFFERENCE IN THE RECON DATA SETS
Explanation: The Multiple Resource Checker utility found no difference in the RECON data sets. This message is printed in the FABWRRT data set.
System action: The Multiple Resource Checker utility continues processing.
User response: None. This message is informational.

FABW0011W  DB TYPES ARE INCONSISTENT. MEMBER member IS NOT COMPARED
Explanation: The Multiple Resource Checker utility did not check the indicated member because the database type, which is defined on the TYPE parameter of the RECON record, did not match. This message is printed in the FABWRRT data set when the utility compares RECON data sets.
System action: The Multiple Resource Checker utility continues processing.
User response: None.

FABW0012W  ACB member IN ACBLIBxx IS NOT COMPARED BECAUSE IT WAS GENERATED BY AN UNSUPPORTED IMS VERSION
Explanation: The indicated ACB member, which was found in the indicated ACB library, was generated by an IMS release that is not supported by the Multiple Resource Checker utility. The RESULT field in the Resource Check Summary report shows DIFF for this ACB member.
System action: The Multiple Resource Checker utility skips this member and continues processing.
User response: None.

FABW0013W  RECONxx DD STATEMENT MISSING
Explanation: The Multiple Resource Checker utility detected an invalid specification for the indicated DD statement.
System action: The Multiple Resource Checker utility continues processing without the indicated DD statement.
User response: Correct the indicated DD statement and rerun the job.

FABW1001E  DUPLICATE CONTROL STATEMENT FOUND
Explanation: Two or more identical control statements were found in the FABWCTL data set.
System action: The Multiple Resource Checker utility ends with a return code of 8.
User response: Remove the duplicate statement and rerun the job.

FABW1002E  INCORRECT PARAMETER FOUND ON THE CONTROL STATEMENT
Explanation: The Multiple Resource Checker utility detected an incorrect parameter in the control statements that are specified in the FABWCTL data set.
System action: The Multiple Resource Checker utility ends with a return code of 8.
User response: Correct the parameter in the control statement and rerun the job.

FABW1003E  INCORRECT MEMBER NAME IS SPECIFIED ON THE CONTROL STATEMENT
Explanation: The member name that is specified on the control statement is incorrect. For example, the member name contained more than 8 characters.
System action: The Multiple Resource Checker utility ends with a return code of 8.
User response: Correct the member name in the control statement and rerun the job.

FABW1004E  INCORRECT CONTROL STATEMENT FOUND
Explanation: A control statement with incorrect format was found in the FABWCTL data set.

System action: The Multiple Resource Checker utility ends with a return code of 8.

User response: Correct the format of the control statement and rerun the job.

FABW1005E CANNOT ACCESS THE RECON DATA SET SPECIFIED BY RECONxxi DD

Explanation: The Multiple Resource Checker utility could not access the RECON data sets because IMS RESLIB is missing or invalid RECON data set is specified.

System action: The Multiple Resource Checker utility ends with a return code of 8.

User response: Ensure that the required libraries are provided with DD statements. Correct the DD statements and rerun the job.

FABW1006E DD STATEMENT FOR resource IS NOT SPECIFIED

Explanation: The DD statement for the indicated resource, which is required to run the function, is missing. One of the following resource names is shown:
- DBD
- PSB
- ACB
- RECON
- IMSVnn
- FABWRRPT

System action: The Multiple Resource Checker utility ends with a return code of 8.

User response: Determine whether the required libraries are supplied with DD statements. Correct the DD statements and rerun the job.

FABW1007E THE NUMBER OF CONTROL STATEMENTS EXCEEDED THE MAXIMUM ALLOWABLE NUMBER

Explanation: The number of control statements in the FABWCTL data set exceeded the maximum value of 9999.

System action: The Multiple Resource Checker utility ends with a return code of 8.

User response: Correct the error and rerun the job.

FABW1008E INCORRECT VERSION OF IMS RESLIB IS SPECIFIED FOR ddbname DD

Explanation: The IMS release level of IMS RESLIB is not consistent with the indicated DD.

System action: The Multiple Resource Checker utility ends with a return code of 8.

User response: Correct the error and rerun the job.

FABW1010E NO MEMBERS ARE PROCESSED

Explanation: The Multiple Resource Checker utility could not find the members to process.

System action: The Multiple Resource Checker utility ends with the return code 8.

User response: Specify the correct libraries or the correct DBD= or PSB= control statement, and rerun the job.

FABW3001E GETMAIN FAILED

Explanation: The GETMAIN macro for storage failed.

System action: The Multiple Resource Checker utility job ends abnormally.

User response: Ensure that the REGION parameter for the JOB or EXEC statement is reasonably large enough. If the region size is small, increase the size, and rerun the job.

FABW3002E OPEN FAILED FOR DDNAME: ddbname

Explanation: The OPEN macro for the indicated DD failed.

System action: The Multiple Resource Checker utility job ends abnormally.

User response: Check whether the correct data set is specified to the DD statement. Correct the error and rerun the job.

FABW3003E LOAD FAILED FOR MODULE: module

Explanation: The LOAD macro for the indicated module failed.

System action: The Multiple Resource Checker utility job ends abnormally.

User response: Check whether the correct load module is contained in the program libraries that are concatenated to the STEPLIB DD statement. Correct the error and rerun the job.

FABW3004E RECON ACCESS FAILED. text

Explanation: An error was detected in the RECON access processing. text provides additional information about the error:
- FUNC=function
- RETURN CODE=return_code
- REASON CODE=reason_code
System action: The Multiple Resource Checker utility job ends abnormally.

User response: Correct the error and rerun the job.

FABW3005E ATTACH FAILED FOR MODULE:
module-name (RC=xx)

Explanation: The ATTACH macro for the indicated module failed.

System action: The Multiple Resource Checker utility job ends abnormally.

User response: Check the return code from ATTACH macro shown in the message, correct the error, and rerun the job.

FABW3006E nnn DIRECTORY READ ERROR

Explanation: A read error occurred when reading the directory. nnn is DBD, PSB or ACB.

System action: The Multiple Resource Checker utility job ends abnormally.

User response: Determine the cause of the READ macro failure, correct the error, and rerun the utility. If the problem persists, contact IBM Software Support.

FABW3007E [RECON | DBD | PSB | ACB(DBD) | ACB(PSB) | CROSS(DBD) | CROSS(PSB) | RECON(DBD)]
COMPARE THREAD ENDS ABNORMALLY

Explanation: The Multiple Resource Checker detected an abnormal thread termination.

System action: The Multiple Resource Checker utility job ends abnormally.

User response: Determine the cause of the thread failure, correct the error, and rerun the utility. If the problem persists, contact IBM Software Support.
FABX messages

Messages that are issued by the Catalog Manager utility begin with the prefix FABX. Also, some messages that are issued when you use the DBD/PSB Map Viewer also begin with the prefix FABX.

FABX0001I CATALOG MANAGER ENDED NORMALLY.
Explanation: The Catalog Manager utility ended successfully.
System action: The Catalog Manager utility ends with a return code of 0.
User response: See the results in the Catalog Manager utility reports.

FABX0002W CATALOG MANAGER ENDED WITH WARNINGS.
Explanation: The Catalog Manager utility detected warning conditions.
System action: The Catalog Manager utility ends with a return code of 4.
User response: For more information about the warning conditions, see other FABX messages and DFS messages that are issued by WTO or messages in the Catalog Manager utility reports.

FABX0003E CATALOG MANAGER ENDED WITH ERRORS.
Explanation: The Catalog Manager utility detected errors.
System action: The Catalog Manager utility ends with a return code of 8.
User response: For more information about the errors, see other FABX messages and DFS messages that are issued by WTO or messages in the Catalog Manager utility reports.

FABX0004W ONE OR MORE INVALID DBD OR PSB RESOURCES FOUND DURING THE ACBGEN TIMESTAMP CHECK PROCESS.
Explanation: The Catalog Manager utility detected inconsistent time stamps between the DBDs and PSBs in the ACB libraries and the DBDs and PSBs in the IMS catalog.
System action: The Catalog Manager utility continues processing.
User response: In the IMS Catalog Validation report, find the DBDs and PSBs that have the INVALID indicator.
To synchronize the DBDs and PSBs in the IMS catalog with the DBD and PSB members in the ACB libraries, run the ACB Generation and Catalog Populate utility (DFS3UACB) or the IMS Catalog Populate utility (DFS3PU00).

FABX0005W VALIDATION FAILED BECAUSE ONE OR MORE DBD OR PSB RESOURCES HAVE NO TIMESTAMP INFORMATION.
Explanation: The time stamps of some resources in the IMS catalog were not found or are invalid. The Catalog Manager utility could not validate some resources.
System action: The Catalog Manager utility continues processing.
User response: In the IMS Catalog Validation report, find the DBDs and PSBs that have the FAILED indicator.
If the failed resource is a PSB that contains PCBs that refer to GSAM databases or logical databases, the Catalog Manager utility does not support the time stamp validation for the PSB because the PSB in the IMS catalog does not have a time stamp.
For other resources, the DBD or PSB resources in the IMS catalog might be corrupted. To correct them, run the ACB Generation and Catalog Populate utility (DFS3UACB) or the IMS Catalog Populate utility (DFS3PU00).

FABX0006W ONE OR MORE DBD OR PSB RESOURCES INCONSISTENT BETWEEN IMS CATALOG AND IMS DIRECTORY.
Explanation: The Catalog Manager utility detected inconsistent time stamps between the DBDs and PSBs in the IMS catalog database and the DBDs and PSBs in the IMS directory data sets. The time stamps of DBDs and PSBs in the IMS catalog database and the IMS directory data sets should always match. The IMS catalog database is corrupted.
System action: The Catalog Manager utility continues processing.
User response: In the IMS Catalog Validation report, find the DBDs and PSBs that have the INVALID indicator.
To repair the IMS catalog, run the ACB Generation and Catalog Populate utility (DFS3UACB) or the IMS Catalog Populate utility (DFS3PU00).
FABX007W • FABX0504E

FABX007W  ONE OR MORE INVALID DBD OR PSB RESOURCES FOUND DURING THE [COMPARE | DECODE] PROCESS.

Explanation: One or more warning messages were issued.
System action: IMS Library Integrity Utilities ends with a return code of 4.
User response: None.

FABX008W  THE PROGRAM VIEW (PSB) IS NOT VALID FOR IMS SQL BECAUSE [DB SCHEMA (DB PCB) IS NOT DEFINED | SCHEMA NAME (PCB NAME) IS NOT DEFINED TO THE DB SCHEMA (DB PCB)].

Explanation: DB Schema (DB PCB) list was not generated because the specified PSB does not have a DB PCB that is effective for IMS SQL.
System action: IMS Library Integrity Utilities ends with a return code of 4.
User response: None.

FABX009W  DBD INSTANCES WITH AN OLD DB VERSION ARE NOT USED BECAUSE DATABASE VERSIONING IS NOT ENABLED.

Explanation: One or more DBDs have multiple DBD instances each with different DB Version (DBVER=). DBD instances with an old DB Version are not used by IMS because database versioning is not enabled in the specified IMS subsystem.
System action: IMS Library Integrity Utilities ends with a return code of 4.
User response: None.

FABX010I  THE CATALOG [VALIDATION | COMPARE] PROCESS IS IN PROGRESS, xxx.x % COMPLETE.

Explanation: This message shows the progress of the validation or the compare process.
System action: IMS Library Integrity Utilities continues processing.
User response: None. This message is informational.

FABX011W  A WARNING MESSAGE IS PRINTED IN THE [DBD | PSB] RESOURCE SOURCE.

Explanation: The DBD or PSB macro source was generated from the indicated resource in the IMS catalog directory. However, a warning was issued. The warning message is printed in the comment line of the DBD or PSB macro source code.
System action: IMS Library Integrity Utilities continues processing.
User response: Locate the warning message in the DBD or PSB macro source code and, if necessary, take necessary steps to resolve the issue.

FABX0501E  UNSUPPORTED LEVEL OF IMS IS BEING USED: xx.x.

Explanation: The version of the IMS.SDFSRESL data set in the STEPLIB is not supported by the Catalog Manager utility.
System action: The Catalog Manager utility ends with a return code of 8.
User response: Specify the IMS.SDFSRESL data set of a supported IMS version and rerun the job.

FABX0502E  ACCESS TO THE IMS CATALOG FAILED.

Explanation: The Catalog Manager utility could not access the IMS catalog database.
System action: The Catalog Manager utility ends with a return code of 8.
User response: Use the DFS messages that were issued during the Catalog Manager utility job step to identify the cause of the error.

FABX0503E  GETMAIN FAILED WITH RC=rc (SIZE=size).

Explanation: The Catalog Manager utility could not obtain enough area when it used the GETMAIN macro.
System action: The Catalog Manager utility ends with a return code of 8.
User response: Increase the value of the REGION= parameter in the JCL. For more information about increasing the value of the JCL REGION parameter, see the topic "REGION parameter" in the z/OS MVS JCL Reference.

FABX0504E  ATTACH FAILED WITH RC=rc, MEMBER member IN ddname DD.

Explanation: The Catalog Manager utility could not attach a subtask.
System action: The Catalog Manager utility ends with a return code of 8.
User response: Ensure that the member or ddname DD data sets are correct and that the data set is not damaged. If the member and the data sets are correct, increase the value of the REGION= parameter in the JCL. For more information about increasing the value of the JCL REGION parameter, see the topic "REGION
**FABX0505E**: OPEN FAILED FOR THE ddname DATA SET. RC=return_code

**Explanation**: The Catalog Manager utility could not open the indicated data set. The return code from the OPEN macro is shown in the message.

**System action**: The Catalog Manager utility ends with a return code of 8.

**User response**: Ensure that the format of the data sets is correct and that the data sets are not damaged. The return code in the message is the return code from the OPEN macro. Use the information about return codes and reason codes in the topic "DESERV completion codes" in z/OS DFSMS Macro Instructions for Data Sets to identify the cause of the error.

---

**FABX0506E**: LOAD FAILED WITH SYSTEM COMPLETION CODE=sc AND RSN=rsn. MEMBER member IN ddname DD.

**Explanation**: The load module member could not be loaded from the indicated data set. The data set might be missing the member, the data set or the load module member might be damaged, or there might be other error causes.

**System action**: The Catalog Manager utility ends with a return code of 8.

**User response**: Use the information about the system completion codes in the topic "System completion codes" in z/OS MVS System Codes to identify the cause of the error.

---

**FABX0507E**: DYNALLOC FAILED FOR THE ddname DD WITH RC=rc AND RSN=rsn. DSN=dsname

**Explanation**: The data set could not be allocated dynamically.

**System action**: The Catalog Manager utility ends with a return code of 8.

**User response**: The return code and reason code in the message are from the DYNALLOC macro. Use the information about the return codes and reason codes in the topic "Interpreting error reason codes from DYNALLOC" in z/OS MVS Programming: Authorized Assembler Services Guide to identify the cause of the error.

---

**FABX0508E**: DESERV FAILED WITH RC=rc AND RSN=rsn. DD=ddname

**Explanation**: The DESERV macro for the indicated DD failed.

---

**FABX0509E**: ddname DD IS MISSING.

**Explanation**: The indicated DD is not allocated.

**System action**: The Catalog Manager utility ends with a return code of 8.

**User response**: See "Control statements for the Catalog Manager utility" on page 342 and specify the indicated DD statement in the JCL.

---

**FABX0510E**: ERROR OCCURRED IN DFSRRC00. COMPLETION CODE IS [USER | SYSTEM] compcode.

**Explanation**: Error occurred in the IMS DFSRRC00 processing.

**System action**: IMS Library Integrity Utilities ends with a return code of 8.

**User response**: If a user completion code is displayed, see the topic "IMS abend codes" in IMS Messages and Codes and correct the error. If a system completion code is displayed, contact IBM Software Support.

---

**FABX0511E**: AN ERROR OCCURRED WHILE READING ACB MEMBER member.

**Explanation**: The indicated member could not be read from the ACB library.

**System action**: The Catalog Manager utility skips the member and continues processing other members. When all of the other members are processed, the Catalog Manager utility ends with a return code of 8.

**User response**: Ensure that the member or the data set directory is not damaged.

---

**FABX0512E**: THERE ARE NO [ACTIVE | PENDING] DBD OR PSB RESOURCES IN THE IMS CATALOG.

**Explanation**: If this message indicates that there are no active DBD or PSB resources in the IMS catalog, it means that the IMS directory data sets are empty. If this message indicates that there are no pending DBD or PSB resources in the IMS catalog, it means that the staging data set is empty.

**System action**: IMS Library Integrity Utilities ends with a return code of 8.
FABX0513E • FABX0523E

User response: If this message is issued for active resources, IMS catalog might not contain correct data. Run the ACB Generation and Catalog Populate utility (DFS3UACB) to correct the IMS catalog. If this message is issued for pending resources, select active resources.

FABX0513E BLDL FOR THE dname DD FAILED WITH RC=rc.

Explanation: The BLDL macro for the indicated DD failed.
System action: The Catalog Manager utility ends with a return code of 8.
User response: Ensure that the format of the data sets is correct and that the data sets are not damaged.

FABX0514E parameter CANNOT BE SPECIFIED FOR THE keyword KEYWORD.

Explanation: An invalid parameter is specified for the keyword.
System action: The Catalog Manager utility ends with a return code of 8.
User response: Correct the keyword parameter and rerun the job. For the supported combinations of keywords and parameters, see “Control statements for the Catalog Manager utility” on page 342.

FABX0516E ERROR IN THE FABXCIN CONTROL STATEMENT.

Explanation: The Catalog Manager utility detected a control statement error in the FABXCIN control statements.
System action: The Catalog Manager utility ends with a return code of 8.
User response: Use other messages with the prefix FABX to identify the error. The messages might be printed in the Catalog Manager utility reports. Correct the error and rerun the job.

FABX0517E statement STATEMENT MUST NOT BE SPECIFIED MORE THAN ONCE.

Explanation: The indicated statement cannot be specified more than once.
System action: The Catalog Manager utility ends with a return code of 8.
User response: Correct the error and rerun the job.

FABX0519E PROC STATEMENT MUST BE SPECIFIED BEFORE THE statement STATEMENT.

Explanation: The PROC statement is not specified on the first line of the FABXCIN control statements.
System action: The Catalog Manager utility ends with a return code of 8.
User response: Specify the PROC statement on the first line in the FABXCIN data set, and then rerun the job.

FABX0520E statement STATEMENT MUST BE SPECIFIED.

Explanation: The indicated statement is not specified in the FABXCIN control statements.
System action: The Catalog Manager utility ends with a return code of 8.
User response: See “Control statements for the Catalog Manager utility” on page 342 and specify the indicated statement. Then, rerun the job.

FABX0521E SYNTAX ERROR IN THE CONTROL STATEMENT.

Explanation: The Catalog Manager utility detected a syntax error in the FABXCIN control statements.
System action: The Catalog Manager utility ends with a return code of 8.
User response: See “Control statements for the Catalog Manager utility” on page 342 and ensure that the control statement conforms to the syntax rules. Then, rerun the job.

FABX0522E UNRECOGNIZED STATEMENT SPECIFIED. STATEMENT: statement

Explanation: The indicated statement is not a valid statement for the Catalog Manager utility.
System action: The Catalog Manager utility ends with a return code of 8.
User response: See “Control statements for the Catalog Manager utility” on page 342 and correct the indicated statement. Then, rerun the job.

FABX0523E keyword KEYWORD CANNOT BE SPECIFIED FOR THE statement STATEMENT.

Explanation: The indicated keyword is not supported for the indicated statement.
System action: The Catalog Manager utility ends with a return code of 8.
User response: See “Control statements for the Catalog Manager utility” on page 342 and correct the control statement. Then, rerun the job.
FABX0524E  THE NUMBER OF keyword KEYWORDS EXCEEDED THE LIMIT. MAX IS num.

Explanation: The number of keywords that are specified in the FABXcin data set exceeds the maximum number of keywords that can be specified.

System action: The Catalog Manager utility ends with a return code of 8.

User response: Specify fewer keywords, and then rerun the job.

FABX0525E  keyword KEYWORD MUST BE SPECIFIED FOR THE statement STATEMENT.

Explanation: The indicated keyword must be specified for the indicated statement.

System action: The Catalog Manager utility ends with a return code of 8.

User response: Add the keyword to the indicated statement, and then rerun the job.

FABX0526E  THE NUMBER OF PARAMETERS SPECIFIED IN keyword KEYWORD EXCEEDED THE LIMIT. MAX IS num.

Explanation: The number of parameters that are specified for the indicated keyword exceeds the maximum number of parameters that can be specified.

System action: The Catalog Manager utility ends with a return code of 8.

User response: Specify fewer parameters, and then rerun the job.

FABX0527E  THE nTH PARAMETER ON THE keyword KEYWORD HAS INCORRECT LENGTH.

Explanation: The length of the indicated parameter that is specified for the indicated keyword is invalid.

System action: The Catalog Manager utility ends with a return code of 8.

User response: See "Control statements for the Catalog Manager utility" on page 342 and correct the length of the indicated parameter. Then, rerun the job.

FABX0531E  MEMBER member WAS GENERATED BY IMS version OR EARLIER.

Explanation: DBD and PSB members in the ACB library cannot be processed because they were generated by an unsupported version of IMS.

System action: IMS Library Integrity Utilities ends with a return code of 8.

User response: Run the ACB Generation and Catalog

FABX0532E  THE RESOURCE resource IN THE [ACBLIB | CATALOG] IS NOT [DBD | PSB].

Explanation: The resource identified by the resource name does not match the specified resource type.

System action: IMS Library Integrity Utilities ends with a return code of 8.

User response: Specify the correct resource name and resource type.

FABX0533W  THE SPECIFIED [DBD | PSB] RESOURCE resource DOES NOT EXIST IN THE [ACBLIB | CATALOG].

Explanation: The specified DBD or PSB resource was not found in the ACBLIB or IMS catalog.

System action: IMS Library Integrity Utilities ends with a return code of 4.

User response: Specify the correct resource name.

FABX0535E  ACCESS TO THE IMS CATALOG FAILED WITH RC=rc AND RSN=rsn. FUNCTION=func.

Explanation: The Catalog Manager utility cannot access the IMS catalog. This message is issued when IMS loads ACBs from the IMS catalog instead of from the ACB libraries and the Catalog Manager utility fails to access the IMS catalog.

System action: The Catalog Manager utility ends with a return code of 8.

User response: Ensure that the following IMS catalog parameters are specified correctly, and then rerun the job:
  - DFSDFxxx member name.
  - Parameters in the FABXPRM DD data set.
  - If you use the Catalog Definition exit routine (DFS3CDX0), the exit routine exists in the STEPLIB concatenation.

If the problem persists, contact IBM Software Support.

FABX0536E  IMS CATALOG IS NOT ENABLED.

Explanation: The IMS catalog is not enabled in the IMS system.

System action: The Catalog Manager utility ends with a return code of 8.

User response: Enable the IMS catalog in one of the following ways:
• By the IMS catalog section of the DFSDF:xxx member that is specified in the FABXPPRM DD data set in the Catalog Manager utility JCL.

• By the IMS Catalog Definition user exit routine (DFS3CDX0) in the STEPLIB DD data set in the Catalog Manager utility JCL.

---

**FABX0537E**  IMS CATALOG IS NOT DEFINED TO THE IMS.

**THE IMS MANAGEMENT OF ACBS IS NOT ENABLED. IMS CATALOG DIRECTORY IS NOT DEFINED TO THE IMS.**

[ACBLIB | CATALOG] IS NOT FOUND IN THE PROCLIB DATA SET CONCATENATION.

**Explanation:** The requested function requires the IMS catalog, IMS catalog directory, or ACB library. However, none of these were found in the specified IMS subsystem.

**System action:** IMS Library Integrity Utilities ends with a return code of 12.

**User response:** Select the correct IMS ID.

---

**FABX0538E**  IMS TOOLS BASE V1.6 OR LATER IS REQUIRED TO ENABLE THE FUNCTION KEYWORD PARAMETERS

**Explanation:** The requested IMS Library Integrity Utilities function requires IBM IMS Tools Base for z/OS Version 1.6 or later, however the load module library of IBM IMS Tools Base V1.6 or later is not found in the STEPLIB DD concatenation.

**System action:** IMS Library Integrity Utilities ends with a return code of 8.

**User response:** Specify the load module library of IBM IMS Tools Base for z/OS V1.6 or later to the STEPLIB DD concatenation.

If you received this message while using Management Console or the ISPF interface of IMS Administration Tool, add the product load module library of IMS Administration Tool to the STEPLIB DD concatenation of the SOT procedure of Distributed Access Infrastructure.

---

**FABX0539E**  CATALOG IS EMPTY.

**Explanation:** No DBDs or PSBs are found in the IMS catalog.

**System action:** IMS Library Integrity Utilities ends with a return code of 8.

**User response:** Ensure that the IMS catalog is correctly defined to the specified IMS.

---

**FABX0541E**  IMS ADMINISTRATION TOOL MODULE IS NOT FOUND IN STEPLIB.

**Explanation:** The product load module library of IMS Administration Tool is not found in the STEPLIB DD concatenation.

**System action:** IMS Library Integrity Utilities ends with a return code of 99.

**User response:** Specify the load module library of IMS Administration Tool to the STEPLIB DD concatenation.

If you received this message while using Management Console or the ISPF interface of IMS Administration Tool, add the product load module library of IMS Administration Tool to the STEPLIB DD concatenation of the SOT procedure of Distributed Access Infrastructure.

---

**FABX0542E**  IMS WAS NOT DISCOVERED. THE IMS MIGHT NOT BE REGISTERED CORRECTLY.

**Explanation:** The specified IMS was not found. The IMS might not be registered to the IMS Tools Knowledge Base repositories.

**System action:** IMS Library Integrity Utilities ends with a return code of 16.

**User response:** Register the IMS ID from the Setup and Administration menu in Management Console or the IMS Administration Tool ISPF dialog.

---

**FABX0543E**  NAME/TOKEN SERVICE FAILED. NAME: name RC=rc.

**Explanation:** Internal error. Name/Token service processing failed.

**System action:** IMS Library Integrity Utilities ends with a return code of 8.

**User response:** Contact IBM Software Support.

---

**FABX0545E**  CATALOG SEARCH INTERFACE PROCESS FOR dsname DATA SET FAILED WITH RC=rc.

**Explanation:** DFSMS Catalog Search Interface (CSI) processing failed.

**System action:** IMS Library Integrity Utilities ends with a return code of 8.

**User response:** Ensure that the IMS catalog is correctly defined.
**FABX0546E** IMS CATALOG DATABASE IS NOT OSAM. DSNAME: dsnname

Explanation: IMS catalog database data sets must be OSAM data sets. The indicated data set is not an OSAM data set.

System action: IMS Library Integrity Utilities ends with a return code of 8.

User response: Ensure that the IMS catalog is correctly defined.

**FABX0547E** UCBSCAN FOR dsnname DATA SET FAILED WITH RC=rc.

Explanation: UCBSCAN processing failed.

System action: IMS Library Integrity Utilities ends with a return code of 8.

User response: Ensure that the IMS catalog is correctly defined.

**FABX0548E** OBTAIN FOR dsnname DATA SET FAILED WITH RC=rc AND DSCB=dscb.

Explanation: OBTAIN processing failed.

System action: IMS Library Integrity Utilities ends with a return code of 8.

User response: Ensure that the IMS catalog is correctly defined.

**FABX0549E** TRKCALC FOR dsnname DATA SET FAILED WITH RC=rc.

Explanation: Internal error. TRKCALC processing failed.

System action: IMS Library Integrity Utilities ends with a return code of 8.

User response: Contact IBM Software Support.

**FABX3901E** OPEN FAILED FOR DATA SET DD: dsnname RC=return_code

Explanation: The specified data set could not be opened.

System action: Processing terminates with a user abend code of U3901.

User response: Ensure that the format of the data set is correct and that the data set is not damaged.

The return code shown in the message is the return code from the OPEN macro. See the return code in the topic "OPEN return codes" in z/OS DFSMS Macro Instructions for Data Sets and correct the error.

**FABX3902E** LOAD FAILED WITH SYSTEM COMPLETION CODE sc AND RSN rsn

(DD: dsnname MEMBER: member)

Explanation: The load module member could not be loaded from the indicated data set.

System action: Processing terminates with a user abend code of U3902.

User response: If the indicated member name is BBES0000 or FABAGVT0, the load module data set of IMS Database Solution Pack or IMS Fast Path Solution Pack is not specified to the STEPLIB. To use the DBD/PSB Map Viewer in Management Console, IMS Library Integrity Utilities must be installed either from IMS Database Solution Pack or IMS Fast Path Solution Pack.

If the member name is not BBES0000 or FABAGVT0, the data set might be missing the member, the data set or the load module member might be damaged, or there might be other error causes. Use the information about the system completion codes in the topic "System completion codes" in z/OS MVS System Codes to identify the cause of the error.

**FABX3903E** GETMAIN FAILED WITH RC=rc (SIZE: size)

Explanation: GETMAIN macro failed.

System action: Processing terminates with a user abend code of U3903.

User response: Increase the value of the REGION= parameter in the SOT (Subordinate Tools Access Server) JCL in a system PROCLIB. SOT is the started task in the IMS Tools Base Distributed Access Infrastructure (DAI).

For more information about SOT, see the topic "Configuring Distributed Access Infrastructure" in the Tools Base Configuration Guide for IMS.

For more information about increasing the JCL REGION parameter, see the topic "REGION parameter" in the z/OS MVS JCL Reference.

**FABX3904E** dsnname DD IS MISSING

Explanation: The indicated DD data set is not specified.

System action: Processing terminates with a user abend code of U3904.

User response: The installation and customization of Management Console or Distributed Access Infrastructure might be incomplete. See IMS Database Solution Pack: Overview and Customization or IMS Fast Path Solution Pack: Overview and Customization and complete the steps to enable the DBD/PSB Map Viewer.
**FABX3906E**  •  **FABX3907E**

**FABX3906E**  THE DATABASE ORGANIZATION
(organization) OF DATABASE dbname IS
NOT SUPPORTED

**Explanation:** The indicated database organization type
is not supported by the DBD/PSB Map Viewer.

**System action:** Processing terminates with job return
code 8.

**User response:** None.

**FABX3907E**  THE LEVEL OF [DB | FP] SOLUTION
PACK IS TOO LOW TO DISPLAY THE
DBD MAP

**Explanation:** The version or the release level of IMS
Database Solution Pack or IMS Fast Path Solution Pack
is lower than the level that is required by the DBD/PSB
Map Viewer.

**System action:** Processing terminates with a user
abend code of U3907.

**User response:** Ensure that the correct STEPLIB load
module data sets are specified. To display the load
module data sets, in Management Console, click Set Up
> Environments, and then check the IMS Library
Integrity Utilities row in the Components table. If
needed, specify the correct load module data set names.
How to look up message explanations

You can use several methods to search for messages and codes.

**Searching for messages on the web**

You can use any of the popular search engines that are available on the web to search for message explanations. When you type the specific message number or code into the search engine, you will be presented with links to the message information in IBM Knowledge Center.
Gathering diagnostic information

Before you report a problem with IMS Library Integrity Utilities to IBM Software Support, you need to gather the appropriate diagnostic information.

Procedure

Provide the following information for all IMS Library Integrity Utilities problems:

- A clear description of the problem and the steps that are required to re-create the problem
- A complete log of the job
- A Load Module/Macro APAR Status report
  
  For information about creating a Load Module/Macro APAR Status report, see “Diagnostics Aid” on page 523.

- The version of IMS that you are using and the version of the operating system that you are using
Diagnostics Aid

If you have a problem that you think is not a user error, use the Diagnostics Aid to collect the necessary information before you contact IBM Software Support.

1. Run Diagnostics Aid (FABLDIAG) and obtain the IMS Library Integrity Utilities Load Module APAR Status report.
2. Attach the report to the other diagnostic documents (such as job dump list or I/O of the utility).
3. Report the error to IBM with the following information:
   - A clear description of the problem and the steps that are required to re-create the problem
   - The version of IMS that you are using and the version of the operating system that you are using
   - A complete log of the job

The Diagnostics Aid generates a Load Module/Macro APAR Status report. This report shows the latest APAR fixes applied to each module and macro.

The Diagnostics Aid is not applicable for any other versions or releases.

How to run Diagnostics Aid with JCL

To run the Diagnostics Aid (FABLDIAG), supply an EXEC statement and DD statements to define input and output data sets.

Use the following JCL example to run the Diagnostics Aid.

```jcl
//stepname EXEC PGM=FABLDIAG
//STEPLIB DD HPS.SHPSLMD0,DISP=SHR
//SHPSLMD DD HPS.SHPSLMD0,DISP=SHR
//SHPSMAC DD HPS.SHPSMAC0,DISP=SHR
//SYSPRINT DD SYSOUT=A
EXEC This statement must have the following form:
   //stepname EXEC PGM=FABLDIAG

JOBLIB or STEPLIB DD
A JOBLIB DD statement or a STEPLIB DD statement must be provided. This statement defines the library containing the FABLDIAG program (usually HPS.SHPSLMD0).

SHPSLMD DD
This statement defines the library containing the load modules (usually HPS.SHPSLMD0) with which you have a problem.

If this DD statement is not provided, or if DD DUMMY is specified, the Load Module APAR Status report is not generated.

It is recommended that you always specify this DD statement.

SHPSMAC DD
This statement defines the library containing the provided macros (usually HPS.SHPSMAC0) for which you have a problem.

If this DD statement is not provided, or if DD DUMMY is specified, the Macro APAR Status report is not generated.

SYSPRINT DD
This output data set contains the Load Module/Macro APAR Status report. The data set contains 133-byte, fixed-length records. It can reside on a tape,
a direct-access device, or a printer; or it can be routed through the output stream. If BLKSIZE is coded in the DD statement, it must be a multiple of 133. However, it is recommended that you use:

//SYSPRINT DD SYSOUT=A

Load Module/Macro APAR Status report

The diagnostics aid generates the Load Module APAR Status report and the Macro APAR Status report. The reports also show the APAR applied to each module and macro most recently.

Load Module APAR Status report

The IMS Library Integrity Utilities Load Module APAR Status report contains information about the modules and their applied APARs.

This report contains the following information:

MODULE LIBRARY

This field includes the data set names specified in the SHPSLMD DD statement. If more than 30 data sets are concatenated, only the first 30 data sets are listed.

MODULE NAME

This field shows the name of the load module member or the alias.

ALIAS-OF

This field shows the name of the original member of the alias. If the module name is not an alias, this field is left blank.

CSECT NAME

This field shows the name of the included CSECT in the module. The CSECT names are reported in the included order in the module.

APAR NUMBER

This field shows the latest APAR number applied to the module represented by the CSECT name. If no APAR is applied, NONE is shown.

APAR FIX-DATE

This field shows the date on which the modification was prepared for the module represented by the CSECT name. If no APAR is applied, N/A is shown.

Notes:

1. If the CSECT name does not start with FAB or HPS, or if the program structure of the CSECT does not conform to IMS Library Integrity Utilities module standard to identify the APAR number and the APAR fixed date, the fields APAR NUMBER and APAR FIX-DATE are filled with asterisks (*).

2. If the load module is a member of the PDSE library, the following statement is shown on the report line, and the job finishes with a return code of 4.

** IT CAN NOT BE ANALYZED DUE TO PDSE LIBRARY MEMBER **

3. If the load macro fails for a utility member, the following statement is shown on the report line and the job completes with a return code of 8.

** IT CAN NOT BE ANALYZED DUE TO LOAD FAILED MEMBER **

Macro APAR Status report

The IMS Library Integrity Utilities Macro APAR Status report contains information about macros and their applied APARs.
This report contains the following information:

**MACRO LIBRARY**
This field includes the data set names specified in the SHPSMAC DD statement. If more than 30 are concatenated, only the first 30 are listed.

**MACRO NAME**
This field shows the name of the macro member or the alias.

**ALIAS-OF**
This field shows the name of the original member of the alias. If the macro name is not an alias, this field is left blank.

**APAR NUMBER**
This field shows the latest APAR number applied to the macro. If no APAR is applied, NONE is shown.

**APAR FIX-DATE**
This field shows the date when the modification was prepared for the macro. If no APAR is applied, N/A is shown.

**Note:** If the macro source statement structure does not conform to IMS Library Integrity Utilities macro standard to identify the APAR number and the APAR fixed date, the fields APAR NUMBER and APAR FIX-DATE are filled with asterisks (*).

### Diagnostic Aid messages and codes

The following topics describe the messages and codes that are issued by the Diagnostics Aid.

#### Return codes

FABLDIAG contains the following return codes:

- **0** The running of the program has been successfully completed.
- **4** Warning messages were issued, but the requested operation was completed.
- **8** Error messages were issued, but the requested operation was completed.

#### Abend codes

All 36xx abend codes are accompanied by an FABU36xx message. Check the appropriate message for problem determination.

#### Messages of the Diagnostics Aid

The FABU messages are issued by the IMS Library Integrity Utilities Diagnostics Aid.

---

**FABU1001I** **DIAG ENDED NORMALLY**

Explanation: This message is generated when Diagnostic Aid has been completed successfully.

System action: Diagnostic Aid completes the job successfully with a return code of 0.

User response: None. This message is informational.

---

**FABU1002W** **DIAG ENDED WITH WARNINGS**

Explanation: This message is generated when Diagnostic Aid encounters trivial error conditions.

---

**FABU1003E** **DIAG ENDED WITH ERRORS**

Explanation: This message is generated when Diagnostic Aid encounters severe error conditions.

System action: Diagnostic Aid ends with a return code of 4.

User response: Check other messages generated by Diagnostic Aid to determine the nature and the cause of the detected errors. Correct the problem, and rerun the job.
FABU1005W • FABU3603E

System action: Diagnostic Aid ends with a return code of 8.

User response: Check other messages generated by Diagnostic Aid to determine the nature and the cause of the errors detected. Correct the problem, and rerun the job.

FABU1005W [SHPSLMD | SHPSMAC] DD STATEMENT NOT FOUND

Explanation: Diagnostic Aid could not find the SHPSLMD/SHPSMAC DD statement.

System action: Diagnostic Aid sets an end-of-job return code of 4 and continues processing. Diagnostic Aid does not generate a report for the load module or the macro.

User response: If you intended to specify the indicated DD statement, correct the error, and rerun the job.

FABU1006W DUPLICATE member IN LIBRARY DDNAME ddname

Explanation: Diagnostic Aid found a duplicated member in the concatenated libraries.

System action: Diagnostic Aid uses the member that is first found in the concatenated libraries. Diagnostic Aid sets an end-of-job return code of 4 and continues processing.

User response: Determine which libraries have the correct module or macro libraries. Correct the error, and if necessary, rerun the job.

FABU1007W DUMMY SPECIFIED FOR [SHPSLMD | SHPSMAC] DD STATEMENT

Explanation: DUMMY was specified for the SHPSLMD/SHPSMAC DD statement.

System action: Diagnostic Aid sets an end-of-job return code of 4 and continues processing. Diagnostic Aid does not generate a report for the load module or the macro.

User response: If you did not intend to specify the dummy DD statement, correct the error, and rerun the job.

FABU1008W NO [MODULE | MACRO] MEMBERS FOUND IN DDNAME [SHPSLMD | SHPSMAC]

Explanation: Diagnostic Aid could not find any utility module or macro members from the DD ddname data set.

System action: Diagnostic Aid sets an end-of-job return code of 4 and continues processing.

User response: Ensure that the libraries have correct utility module or macro libraries. Correct the error, and rerun the job.

FABU2001E LOAD FAILED FOR DDNAME ddname

MODULE member

Explanation: Diagnostic Aid could not load a member from ddname.

System action: Diagnostic Aid sets an end-of-job return code of 4 and continues processing.

User response: Ensure that the member indicated exists in the data set specified for the indicated ddname. Correct the error, and rerun the job.

FABU3600E OPEN FAILED FOR DDNAME ddname

Explanation: The named DCB could not be opened.

System action: Diagnostic Aid ends with an abend code of U3600.

User response: Ensure that a ddname DD statement exist, and that it specifies the correct DD parameter. Correct any errors, and rerun the job.

FABU3601E GET FAILED FOR DDNAME ddname

Explanation: The GET failed for a directory from the DD ddname data set.

System action: Diagnostic Aid ends with an abend code of U3601.

User response: See the MVS system message and its programmer response. Correct the error, and rerun Diagnostic Aid. If the error persists, contact IBM Software Support.

FABU3602E READ FAILED FOR DDNAME ddname

MEMBER member

Explanation: The READ failed for a member from the DD ddname data set.

System action: Diagnostic Aid ends with an abend code of U3602.

User response: See the MVS system message and its programmer response. Correct the error, and rerun Diagnostic Aid. If the error persists, contact IBM Software Support.

FABU3603E BLDL FAILED FOR DDNAME ddname

MEMBER member

Explanation: The member was not found when the BLDL macro searched the PDS directory for the ddname.

System action: Diagnostic Aid ends with an abend code of U3603.

User response: Ensure that the member indicated exists in the data set specified for the indicated
ddname. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABU3604E  LOAD FAILED FOR DDNAME  ddname  MODULE  member

Explanation:  Diagnostic Aid could not load member name from the ddname.

System action:  Diagnostic Aid ends with an abend code of U3604.

User response:  See the MVS system message and its programmer response. Correct the error, and rerun Diagnostic Aid. If the error persists, contact IBM Software Support.

FABU3605E  DELETE FAILED FOR MODULE  member

Explanation:  Diagnostic Aid could not delete a member name.

System action:  Diagnostic Aid ends with an abend code of U3605.

User response:  Contact IBM Software Support.

FABU3606E  PUT FAILED FOR SYSPRINT

Explanation:  Diagnostic Aid could not put report data in SYSPRINT.

System action:  Diagnostic Aid ends with an abend code of U3606.

User response:  See the MVS system message and its programmer response. Correct the error, and rerun Diagnostic Aid. If the error persists, contact IBM Software Support.

FABU3607E  OPEN FAILED FOR SYSPRINT

Explanation:  SYSPRINT DCB could not be opened.

System action:  Diagnostic Aid ends with an abend code of U3607.

User response:  Ensure that a ddname SYSPRINT DD statement exists, and that it specifies the correct DD parameter. Correct any errors, and rerun the job.

FABU3608E  FIND FAILED FOR DDNAME  ddname  MEMBER  member

Explanation:  The FIND failed for a member from DDNAME ddname data set.

System action:  Diagnostic Aid ends with an abend code of U3608.

User response:  Ensure that the member indicated exists in the data set specified for the indicated ddname. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

FABU3609E  DEVTYPE FAILED FOR DDNAME  ddname

Explanation:  The DEVTYPE failed for a DDNAME ddname data set.

System action:  Diagnostic Aid ends with an abend code of U3609.

User response:  Contact IBM Software Support.

FABU3610E  RDJFCB FAILED FOR DDNAME  ddname

Explanation:  The READJFCB failed for a DDNAME ddname data set.

System action:  Diagnostic Aid ends with an abend code of U3610.

User response:  Contact IBM Software Support.

FABU3611E  GETMAIN FAILED, INSUFFICIENT STORAGE TO RUN THE JOB

Explanation:  Work space for Diagnostic Aid could not be obtained.

System action:  Diagnostic Aid ends with an abend code of U3611.

User response:  Increase the region size, and rerun the job.

FABU3612E  TOO MANY [MODULE | MACRO] MEMBERS DETECTED IN DDNAME [SFABMOD | SHPSMAC]

Explanation:  There are too many utility members in the SFABMOD data set or in the SHPSMAC DD data set.

System action:  Diagnostic Aid ends with an abend code of U3612.

User response:  Specify the correct data set for the indicated DD statement, and rerun the job.
Chapter 15. References

The following reference topics provide technical reference information for using IMS Library Integrity Utilities.

Topics:

- “Device and feature code tables” on page 530
- “Sample library members” on page 532
- “How to read syntax diagrams” on page 534
Device and feature code tables

The following tables list the device codes and the feature codes that are associated with various devices.

## Device code table

*Table 33. Device code table*

<table>
<thead>
<tr>
<th>Device code</th>
<th>Device</th>
<th>Device code</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>3270,1</td>
<td>21</td>
<td>DPM-B01</td>
</tr>
<tr>
<td>01</td>
<td>3270P,1</td>
<td>22</td>
<td>DPM-B02</td>
</tr>
<tr>
<td>02</td>
<td>3270,2</td>
<td>23</td>
<td>DPM-B03</td>
</tr>
<tr>
<td>03</td>
<td>3270P,2</td>
<td>24</td>
<td>DPM-B04</td>
</tr>
<tr>
<td>04</td>
<td>274X</td>
<td>25</td>
<td>DPM-B05</td>
</tr>
<tr>
<td>05</td>
<td>FIDS</td>
<td>26</td>
<td>DPM-B06</td>
</tr>
<tr>
<td>06</td>
<td>FIDS3</td>
<td>27</td>
<td>DPM-B07</td>
</tr>
<tr>
<td>07</td>
<td>FIDS4</td>
<td>28</td>
<td>DPM-B08</td>
</tr>
<tr>
<td>08</td>
<td>FIN</td>
<td>29</td>
<td>DPM-B09</td>
</tr>
<tr>
<td>09</td>
<td>FIJP</td>
<td>2A</td>
<td>DPM-B10</td>
</tr>
<tr>
<td>0A</td>
<td>FIPB</td>
<td>2B</td>
<td>DPM-B11</td>
</tr>
<tr>
<td>0B</td>
<td>FIFP</td>
<td>2C</td>
<td>DPM-B12</td>
</tr>
<tr>
<td>0C</td>
<td>SCS1</td>
<td>2D</td>
<td>DPM-B13</td>
</tr>
<tr>
<td>0D</td>
<td>SCS2</td>
<td>2E</td>
<td>DPM-B14</td>
</tr>
<tr>
<td>0E</td>
<td>FIDS7</td>
<td>2F</td>
<td>DPM-B15</td>
</tr>
<tr>
<td>11</td>
<td>DPM-A01</td>
<td>41</td>
<td>3270-A01</td>
</tr>
<tr>
<td>12</td>
<td>DPM-A02</td>
<td>42</td>
<td>3270-A02</td>
</tr>
<tr>
<td>13</td>
<td>DPM-A03</td>
<td>43</td>
<td>3270-A03</td>
</tr>
<tr>
<td>14</td>
<td>DPM-A04</td>
<td>44</td>
<td>3270-A04</td>
</tr>
<tr>
<td>15</td>
<td>DPM-A05</td>
<td>45</td>
<td>3270-A05</td>
</tr>
<tr>
<td>16</td>
<td>DPM-A06</td>
<td>46</td>
<td>3270-A06</td>
</tr>
<tr>
<td>17</td>
<td>DPM-A07</td>
<td>47</td>
<td>3270-A07</td>
</tr>
<tr>
<td>18</td>
<td>DPM-A08</td>
<td>48</td>
<td>3270-A08</td>
</tr>
<tr>
<td>19</td>
<td>DPM-A09</td>
<td>49</td>
<td>3270-A09</td>
</tr>
<tr>
<td>1A</td>
<td>DPM-A10</td>
<td>4A</td>
<td>3270-A10</td>
</tr>
<tr>
<td>1B</td>
<td>DPM-A11</td>
<td>4B</td>
<td>3270-A11</td>
</tr>
<tr>
<td>1C</td>
<td>DPM-A12</td>
<td>4C</td>
<td>3270-A12</td>
</tr>
<tr>
<td>1D</td>
<td>DPM-A13</td>
<td>4D</td>
<td>3270-A13</td>
</tr>
<tr>
<td>1E</td>
<td>DPM-A14</td>
<td>4E</td>
<td>3270-A14</td>
</tr>
<tr>
<td>1F</td>
<td>DPM-A15</td>
<td>4F</td>
<td>3270-A15</td>
</tr>
</tbody>
</table>
## Feature code table

Table 34. Feature code table

<table>
<thead>
<tr>
<th>Feature code</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>FEAT=1</td>
</tr>
<tr>
<td>02</td>
<td>FEAT=2</td>
</tr>
<tr>
<td>03</td>
<td>FEAT=3</td>
</tr>
<tr>
<td>04</td>
<td>FEAT=4</td>
</tr>
<tr>
<td>05</td>
<td>FEAT=5</td>
</tr>
<tr>
<td>06</td>
<td>FEAT=6</td>
</tr>
<tr>
<td>07</td>
<td>FEAT=7</td>
</tr>
<tr>
<td>08</td>
<td>FEAT=8</td>
</tr>
<tr>
<td>09</td>
<td>FEAT=9</td>
</tr>
<tr>
<td>0A</td>
<td>FEAT=10</td>
</tr>
<tr>
<td>40</td>
<td>FEAT=120</td>
</tr>
<tr>
<td>4A</td>
<td>FEAT=(NOCD,DEKYBD,PEN)</td>
</tr>
<tr>
<td>4B</td>
<td>FEAT=(CARD,DEKYBD,PEN)</td>
</tr>
<tr>
<td>50</td>
<td>FEAT=126</td>
</tr>
<tr>
<td>60</td>
<td>FEAT=132</td>
</tr>
<tr>
<td>7F</td>
<td>FEAT=IGNORE</td>
</tr>
<tr>
<td>C1</td>
<td>FEAT=(CARD,NOPFK,NOPEN)</td>
</tr>
<tr>
<td>C2</td>
<td>FEAT=(NOCD,NOPFK,PEN)</td>
</tr>
<tr>
<td>C3</td>
<td>FEAT=(CARD,NOPFK,PEN)</td>
</tr>
<tr>
<td>C4</td>
<td>FEAT=(NOCD,PFK,NOPEN)</td>
</tr>
<tr>
<td>C5</td>
<td>FEAT=(NOCD,PFK,NOPEN)</td>
</tr>
<tr>
<td>C6</td>
<td>FEAT=(NOCD,PFK,PEN)</td>
</tr>
<tr>
<td>C7</td>
<td>FEAT=(CARD,PFK,PEN)</td>
</tr>
<tr>
<td>C8</td>
<td>FEAT=(NOCD,DEKYBD,NOPEN)</td>
</tr>
<tr>
<td>C9</td>
<td>FEAT=(CARD,DEKYBD,NOPEN)</td>
</tr>
</tbody>
</table>
Sample library members

The sample libraries (SHPSJCL0 and SHPSSAMP) that are supplied with IMS Library Integrity Utilities contains JCL that you can use as a model to create your own jobs.

The following table summarizes the members in the SHPSJCL0 library.

Table 35. Sample JCL in the SHPSJCL0 library

<table>
<thead>
<tr>
<th>Utility</th>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All utilities</td>
<td>FABLLINK</td>
<td>Link-edits the IMS Library Integrity Utilities load modules.</td>
</tr>
<tr>
<td>Integrity Checker</td>
<td>FABLIVP3</td>
<td>Runs the Integrity Checker utility in the IMS batch environment to create an RDE. Before running this JCL, make sure that Integrity Checker is activated.</td>
</tr>
<tr>
<td></td>
<td>FABLALSC</td>
<td>Creates alias name DSPCRTR0 for the FABLRTR0 module.</td>
</tr>
<tr>
<td></td>
<td>FABLALSD</td>
<td>Deletes the alias name DSPCRTR0 from the FABLRTR0 module.</td>
</tr>
<tr>
<td></td>
<td>FABLINIT</td>
<td>Initializes the Integrity Checker utility by creating a LICON data set, initializing the LICON data set, and creating a global option module.</td>
</tr>
<tr>
<td></td>
<td>FABLUMD1</td>
<td>Runs SMP/E RECEIVE/APPLY of USERMOD to install the FABLRTR0 module into the IMS SDFSRESL library.</td>
</tr>
<tr>
<td>Consistency Checker</td>
<td>FABLIVP2</td>
<td>Runs the Consistency Checker utility.</td>
</tr>
<tr>
<td>Multiple Resource Checker</td>
<td>FABWIVP</td>
<td>Runs the Multiple Resource Checker utility.</td>
</tr>
<tr>
<td>DBD/PSB/ACB Compare, Mapper, Reversal</td>
<td>FABLIVP1</td>
<td>Runs the DBD/PSB/ACB Compare, Mapper, and Reversal utilities.</td>
</tr>
<tr>
<td>Catalog Manager</td>
<td>FABXCIVP</td>
<td>Runs the Catalog Manager utility.</td>
</tr>
<tr>
<td>Advanced ACBGEN and ACBLIB Analyzer</td>
<td>FABQIVP</td>
<td>Runs the Advanced ACBGEN utility and the ACBLIB Analyzer utility.</td>
</tr>
<tr>
<td></td>
<td>FABLQUMD1</td>
<td>Deletes alias DFSUACB0 from the SHPSLMD0 library of IMS Library Integrity Utilities and the LMOD entry of IMS Library Integrity Utilities SMP/E CSI.</td>
</tr>
<tr>
<td></td>
<td>FABLQUMD2</td>
<td>Runs SMP/E LIST for the IMS DFSRRA80 source entry.</td>
</tr>
<tr>
<td></td>
<td>FABLQUMD3</td>
<td>Runs SMP/E RECEIVE/APPLY of USERMOD to modify the IMS DFSRRA80 module so that the module invokes the FABQMAIN module of IMS Library Integrity Utilities instead of DFSUACB0.</td>
</tr>
<tr>
<td>MFS Reversal and Compare</td>
<td>FABVIVP</td>
<td>Runs the MFS Reversal utility and the MFS Compare utility.</td>
</tr>
</tbody>
</table>

The following tables summarize the members in the SHPSSAMP library.
### Table 36. Sample JCL in the SHPSSAMP library

<table>
<thead>
<tr>
<th>Utility</th>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity Checker</td>
<td>FABLCNV2</td>
<td>Migrates the LICON data set that is used in IMS Library Integrity Utilities Version 1 Release 1.</td>
</tr>
<tr>
<td>DBD/PSB/ACB Reversal</td>
<td>FABNDFL1</td>
<td>Runs the DBD/PSB/ACB Reversal Site Default Generation utility to generate a SYSIN site default table.</td>
</tr>
<tr>
<td></td>
<td>FABNDFL2</td>
<td>Runs the DBD/PSB/ACB Reversal Site Default Generation utility to report on the SYSIN site default table.</td>
</tr>
</tbody>
</table>

### Table 37. Sample procedure in the SHPSSAMP library

<table>
<thead>
<tr>
<th>Utility</th>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity Checker</td>
<td>FABLPGEN</td>
<td>This member contains the procedure for creating a global option module for the Integrity Checker utility.</td>
</tr>
</tbody>
</table>
How to read syntax diagrams

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

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

- Required items appear on the horizontal line (the main path).

```
>>>required_item
```

- Optional items appear below the main path.

```
>>>required_item
| optional_item |
```

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

```
>>>required_item
| optional_item |
```

- If you can choose from two or more items, they appear vertically, in a stack. If you must choose one of the items, one item of the stack appears on the main path.

```
>>>required_item
| required_choice1
| required_choice2 |
```

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

```
>>>required_item
| optional_choice1
| optional_choice2 |
```

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

```
>>>required_item
| default_choice
| optional_choice
| optional_choice |
```

- An arrow returning to the left, above the main line, indicates an item that can be repeated.
If the repeat arrow contains a comma, you must separate repeated items with a comma.

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

- Keywords, and their minimum abbreviations if applicable, appear in uppercase. They must be spelled exactly as shown. Variables appear in all lowercase italic letters (for example, column-name). They represent user-supplied names or values.
- Separate keywords and parameters by at least one space if no intervening punctuation is shown in the diagram.
- Enter punctuation marks, parentheses, arithmetic operators, and other symbols exactly as shown in the diagram.
- Footnotes are shown by a number in parentheses; for example, (1).
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 give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

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 supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM 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 information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

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.

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 http://www.ibm.com/legal/copytrade.shtml.

Other company, product, and service names may be trademarks or service marks of others.

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.

**Privacy policy considerations**

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

DBD (continued)
library information (DBD/PSB/ACB Reversal) 302
map (DBD/PSB/ACB Mapper) 261
report (DBD/PSB/ACB Mapper) 268
segment information (DBD/PSB/ACB Mapper) 274
DBD Check report (Consistency Checker) 141
DBD Compare report (DBD/PSB/ACB Compare) 227
DBD keyword (Multiple Resource Checker) 175
DBD map (DBD/PSB/ACB Mapper) 261
DBD report (DBD/PSB/ACB Mapper) 268
DBD XREF by Access report
DBD XREF by Access - Access Order
(DBD/PSB/ACB Reversal) 309
DBD XREF by Access - DBD Name Order
(DBD/PSB/ACB Reversal) 309
DBD, PSB, and ACB map functions 253
DBD/PSB Map Viewer 2, 15
DBD/PSB Names Specified via SYSIN
report (Advanced ACBGEN) 366
DBD/PSB/ACB Compare 199
examples 223, 227
FABLCOMP 200
input 207
ACB control statement 209
DBD control statement 209
NOCOMP control statement 213
PSB control statement 209
REPORT control statement 212
SYSIN data set 207
JCL requirements 204
job steps 200
output 227
SYSOUT data set 227
SYSPRINT data set 227
overview 200
run 203
DBD/PSB/ACB Mapper 243, 244
examples 253
FABMMAIN 244
input 250
SYSIN data set 250
JCL requirements 248
job steps 244
output 259, 265
ACB (DBD) map 261
ACB (DBD) report 268
ACB (PSB) map 261
ACB (PSB) report 268
ACB (PSB) Summary report 265
DBD map 261
DBD report 268
PSB map 261
PSB report 268
PSB Summary report 265
SYSPRINT data set 259, 268
overview 244
run 247
DBD/PSB/ACB Reversal 279
examples 300
DBD/PSB/ACB Reversal (continued)
FABNRVS 280
input 288
SYSIN data set 288
JCL requirements 285
output
DBDSRC data set 308
MAPOUT data set 325
OPTPTPT data set 325
PSBSRC data set 308
SYSPUT data set 304
SYSPRINT data set 309
SYSPUNCH data set 305
overview 280
run 284
Site Default Generation utility 326
overview 326
DBDLIB DD
Consistency Checker 131
DBD/PSB/ACB Compare 204
DBD/PSB/ACB Mapper 248
DBD/PSB/ACB Reversal 285
LICON utility 98
DBDLIB2 DD (DBD/PSB/ACB Compare) 204
DBDLIBxx DD (Multiple Resource Checker) 172
DBDSRC DD (DBD/PSB/ACB Reversal) 308
DBDSRC DD (DBD/PSB/ACB Reversal) 285
DBRRC procedures 60
DBTSNAP DD (ACBLIB Analyzer) 382
DD statements
ACBCATWK DD (Advanced ACBGEN) 335
ACBLIB DD
Catalog Manager 338
Consistency Checker 131
DBD/PSB/ACB Compare 204
DBD/PSB/ACB Mapper 248
DBD/PSB/ACB Reversal 285
LICON utility 98
ACBLIB2 DD (DBD/PSB/ACB Compare) 204
ACBLIBxx DD (Multiple Resource Checker) 172
ACBSYSIN DD
ACBLIB Analyzer 382
Advanced ACBGEN 355
COMPCTL DD (Advanced ACBGEN) 355
COPYFMT DD (MFS Reversal) 401
COPYRPT DD (MFS Reversal) 401
DBDLIB DD
Consistency Checker 131
DBD/PSB/ACB Compare 204
DBD/PSB/ACB Mapper 248
DBD/PSB/ACB Reversal 285
LICON utility 98
DBDLIB2 DD (DBD/PSB/ACB Compare) 204
DBDLIBxx DD (Multiple Resource Checker) 172
DBDSRC DD (DBD/PSB/ACB Reversal) 285
DBTSNAP DD (ACBLIB Analyzer) 382
DD statements (continued)
DBTSNAP DD (ACBLIB Analyzer) 382
DFS3PPRM DD (Advanced ACBGEN) 355
DFSHPDSC DD (Catalog Manager) 338
DFSMDA DD (Consistency Checker) 131
DFSPRINT DD (Advanced ACBGEN) 355
DFSRESLBD DD
Advanced ACBGEN 355
Catalog Manager 338
Consistency Checker 131
LICON utility 98
DFSVSAMP DD (Catalog Manager) 338
FABLICON DD (LICON utility) 98
FABLIN DD (LICON utility) 98
FABLPRT DD
Integrity Checker 60
LICON utility 98
FABLSNAP DD (Integrity Checker) 60
FABQRPT DD (ACBLIB Analyzer) 382
FABWCTL DD (Multiple Resource Checker) 172
FABWOUT DD (Multiple Resource Checker) 172
FABWRRPT DD (Multiple Resource Checker) 172
FABWSUMM DD (Multiple Resource Checker) 172
FABXCIN DD (Catalog Manager) 338
FABXCRP0 DD (Catalog Manager) 338
FABXCRP1 DD (Catalog Manager) 338
FABXPPRM DD (Catalog Manager) 338
FORMAT DD
MFS Compare 429
MFS Reversal 401
FORMAT2 DD (MFS Compare) 429
IMS DD
Advanced ACBGEN 355
Catalog Manager 338
IMSCACB DD
ACBLIB Analyzer 382
Advanced ACBGEN 355
IMSCACB01 DD (Advanced ACBGEN) 355
IMSVnmm DD (Multiple Resource Checker) 172
JOBLIB DD
Catalog Manager 338
DBD/PSB/ACB Compare 204
DBD/PSB/ACB Mapper 248
DBD/PSB/ACB Reversal 285
LICON utility 98
Multiple Resource Checker 172
MAPOUT DD (DBD/PSB/ACB Reversal) 285
DD statements (continued)
MFSRCE DD
  MFS Compare 429
  MFS Reversal 401
MFSRCE2 DD (MFS Compare) 429
MODBLKS DD (Consistency Checker) 131
NSYSRDDS DD (Consistency Checker) 131
OPTPTD DD (DBD/PSB/ACB Reversal) 285
PROCLIB DD
  Advanced ACBGEN 355
  Catalog Manager 338
PSLIB DD
  Consistency Checker 131
  DBD/PSB/ACB Compare 204
  DBD/PSB/ACB Mapper 248
  DBD/PSB/ACB Reversal 285
PSLIB2 DD (DBD/PSB/ACB Compare) 204
PSLIBxx DD (Multiple Resource Checker) 172
PSBSRC DD (DBD/PSB/ACB Reversal) 285
RECONx DD
  Catalog Manager 338
  Consistency Checker 131
  LICON utility 98
RECONxxn DD (Multiple Resource Checker) 172
SORT DD (ACBLIB Analyzer) 382
STEPLIB DD
  ACBLIB Analyzer 382
  Advanced ACBGEN 355
  Catalog Manager 338
  Consistency Checker 131
  DBD/PSB/ACB Compare 204
  DBD/PSB/ACB Mapper 248
  DBD/PSB/ACB Reversal 285
  Integrity Checker 60
  LICON utility 98
  MFS Compare 429
  MFS Reversal 401
  Multiple Resource Checker 172
SYSABEND DD (Catalog Manager) 338
SYSIN DD
  Advanced ACBGEN 355
  Consistency Checker 131
  DBD/PSB/ACB Compare 204
  DBD/PSB/ACB Mapper 248
  DBD/PSB/ACB Reversal 285
  FABLPGEN program 92
  MFS Compare 429
  MFS Reversal 401
SYSLIB DD (FABLPGEN program) 92
SYSLMOD DD (FABLPGEN program) 92
SYSDMMDUMP DD (Catalog Manager) 338
SYSOUT DD
  Consistency Checker 131
  DBD/PSB/ACB Compare 204
  DBD/PSB/ACB Mapper 248
  DBD/PSB/ACB Reversal 285
DD statements (continued)
SYSOUT DD (continued)
  MFS Reversal 401
SYSPRNT DD
  ACBLIB Analyzer 382
  Advanced ACBGEN 355
  Consistency Checker 131
  DBD/PSB/ACB Compare 204
  DBD/PSB/ACB Mapper 248
  DBD/PSB/ACB Reversal 285
  MFS Compare 429
  MFS Reversal 401
  Multiple Resource Checker 172
SYSPUNCH DD (DBD/PSB/ACB Reversal) 285
SYSTRDSS DD (Consistency Checker) 131
SYSDUMP DD (Catalog Manager) 338
SYSTUD DD (Advanced ACBGEN) 355
SYSTU4 DD (Advanced ACBGEN) 355
SYSTOUT DD
  MFS Compare 429
  DEFN option (MFS Reversal) 417
  DELETE control statement (Advanced ACBGEN) 359
  DELETE.DB command (LICON utility) 110
  device characteristics table (MFS Reversal) 398, 403
  device code table 530
  DFLD label (MFS Reversal) 410, 417
  DFLDNAME keyword (MFS Compare) 423
  DFS Messages Summary report (Advanced ACBGEN) 372
  DFS3PPRM DD (Advanced ACBGEN) 355
  DFSHDBSBC DD (Catalog Manager) 338
  DFSMDA DD (Consistency Checker) 131
  DFSnnnn messages (Advanced ACBGEN) 355
  DFSPRINT DD (Advanced ACBGEN) 355
  DFSRESLB DD
    Advanced ACBGEN 355
    Catalog Manager 338
    Consistency Checker 131
    LICON utility 98
    MFS Compare 429
    MFS Reversal 401
    EXIT keyword (MFS Compare) 423
    EXPIRE.DB command (LICON utility) 112
Distribution of PSB Workarea Sizes report (ACBLIB Analyzer) 393
DIV statement (MFS Reversal) 416
DMB verification restrictions 79
DMB verification option 80
documentation accessing 21
sending feedback 21
double-step verification 80, 92
DPAGE label (MFS Reversal) 410
DPMA devices (MFS Reversal) 416
DPMB devices (MFS Reversal) 416
DPN keyword (MFS Compare) 425
DSCA keyword (MFS Compare) 424
DSCA value (MFS Reversal) 416
DVCBL control statement
  MFS Compare 431
  MFS Reversal 403

E
EATTR
  MFS Compare 426
  MFS Reversal 416
EGCS specification (MFS Reversal) 416
EJECT option (MFS Reversal) 417
ENDMSG (MFS Reversal) 417
examples
  Catalog Manager 342
  Consistency Checker 137
  DBD/PSB/ACB Compare 223, 227
  DBD/PSB/ACB Mapper 253
  DBD/PSB/ACB Reversal 300
  Multiple Resource Checker 182
EXCLUDE control statement
  MFS Reversal 403
EXEC statement
  ACBLIB Analyzer 382
  Advanced ACBGEN 355
  Catalog Manager 338
  DBD/PSB/ACB Compare 204
  DBD/PSB/ACB Mapper 248
  DBD/PSB/ACB Reversal 285
  LICON utility 98
  MFS Compare 429
  MFS Reversal 401
EXIT keyword (MFS Compare) 423
EXPIRE.DB command (LICON utility) 112

F
FABCOMP (Compare utility) 199
FABLECHK (Consistency Checker) 126
FABLIBON DD (LICON utility) 98
FABLID DD (LICON utility) 98
FABLPGEN control statement keywords 92
FABLPGEN JCL requirements 92
FABLPGEN procedure 54, 91
FABLPGEN macro 92
FABLPRT DD
  Integrity Checker 60
  LICON utility 98
FABLSNAP DD (Integrity Checker) 60
IMS Library Integrity Utilities User's Guide

**G**
G’LITERAL’ keyword (MPS Compare) 426
GENMAX keyword (LICON utility) 104
global option module creating 91
global option module generation macro 91
global option modules creating 54GRAPHIC keyword (MPS Compare) 423

**H**
hardware prerequisites 26
HDRCTL (MPS Reversal) 416
HDRCTL keyword (MPS Compare) 425
HTAB keyword (MPS Compare) 424

**I**
IF label (MPS Reversal) 410
IMS 6
IMS DD Advanced ACBGEN 355
IMS DD (Catalog Manager) 338
IMS messages 445
IMSACB DD ACBLIB Analyzer 382
Advanced ACBGEN 355
IMSACB01 DD (Advanced ACBGEN) 355
IMSVnn DD (Multiple Resource Checker) 172
INDD= operand (ACBLIB Analyzer) 385
INIT=DB command (LICON utility) 101
INIT=LICON command (LICON utility) 106
input ACBLIB Analyzer 384
Advanced ACBGEN 359
Catalog Manager 342
Consistency Checker 134
DBD/PSB/ACB Compare 207
DBD/PSB/ACB Mapper 250
DBD/PSB/ACB Reversal 288
LICON utility 100
MFS Compare 431
MFS Reversal 403
Multiple Resource Checker 175
Input Specifications report (ACBLIB Analyzer) 386
Input Specifications report (Advanced ACBGEN) 363
Integrity Checker 28, 35 activating 54
addressing DMB mismatch 84
applying PTFs 78
BPE-based DBRC 65
configuration 41
considerations 50
database access recording option 49
deactivating 86
design configuration 39
global option modules 40, 54
changing 75
effective ranges 76
historical data in LICON data sets 49
LICON data sets 40, 55
LIU load module library 39
LIU load modules 60
maintaining 67
maintaining global option modules 75
maintaining LICON data sets 77
maintaining RDEs 67
database recovery 70
database reorganization 68
DBD change 71
initial database load 67
overview 36
planning 39
preventing 79
program structures 36
RACF security 58
restart 66, 77
restrictions 79
runtime options 47
verifying activation 66

**J**
JCL requirements (continued)
DBD/PSB/ACB Compare 204
DBD/PSB/ACB Mapper 248
DBD/PSB/ACB Reversal 285
FABLIU00 98
FABLPGEN 92
LICON utility 98
MFS Compare 429
MFS Reversal 401
Multiple Resource Checker 172
JOBLIB DD Catalog Manager 338
DBD/PSB/ACB Compare 204
DBD/PSB/ACB Mapper 248
DBD/PSB/ACB Reversal 285
LICON utility 98
Multiple Resource Checker 172
JUST keyword (MPS Compare) 423

**K**
keyboard shortcuts 23

**L**
LDEL keyword (MPS Compare) 424
legal notices cookie policy 537, 539
notices 537
programming interface information 537
trademarks 537, 538
LENGTH keyword (MPS Compare) 427
libraries used (ACBLIB Analyzer) 387
Library Contents report (ACBLIB Analyzer) 388
library information of DBD (DBD/PSB/ACB Reversal) 302
library information of PSB (DBD/PSB/ACB Reversal) 302
Library Information report (ACBLIB Analyzer) 387
LIBTYPE= operand (ACBLIB Analyzer) 384
LICON data sets defining and initializing 56
estimating size 51
LICON utility 98
control statements 98
input 100
JCL requirements 98
output 118
LICON utility control statement keywords 98
LIST=DB command (LICON utility) 113
LIST=LICON command (LICON utility) 114
LISTLIB command (ACBLIB Analyzer) 384
LITERAL keyword (MPS Compare) 426
LIU load module library 39
Load Module Mgmt Stats report (Advanced ACBGEN) 376
lower case characters (MFS Reversal) 416
LTH keyword (MPS Compare) 423, 426
migration (continued)

Advanced ACBGEN 33
Miscellaneous DFS Messages report (Advanced ACBGEN) 371
MODBLKS DD (Consistency Checker) 131
MODE keyword (MFS Compare) 424
MONITOR= operand (Advanced ACBGEN) 359
MULT keyword (MFS Compare) 425
Multiple Resource Checker 157
checking consistencies 160
control statements 175
examples 182
JCL requirements 172
output 190
FABWOUT data set 190
FABWRPRT data set 194
FABWSUMM data set 190
RECON Difference report 194
Resource Check Summary report 190
overview 138

output (continued)

MFS Compare 433
MFS Reversal 417
Multiple Resource Checker 190

P

PAGE keyword (MFS Compare) 423
PAGE keyword (MFS Reversal) 417
page size (MFS Reversal) 416
PAGESIZE= operand (Advanced ACBGEN) 359
PAGINGOP keyword (MFS Compare) 426
PASSWORD keyword (MFS Compare) 423, 426
PD keyword (MFS Compare) 425
PDB keyword (MFS Compare) 424
PEN keyword (MFS Compare) 424, 426
PEN name (MFS Reversal) 410
PFK keyword (MFS Compare) 424
PFK name (MFS Reversal) 410
PGE keyword (MFS Compare) 424
PID keyword (MFS Compare) 426
POS keyword (MFS Compare) 426
post-compression (Advanced ACBGEN) 365
PPAGE keyword (MFS Compare) 425
PPAGE name (MFS Reversal) 416
PPAGE statement (MFS Reversal) 416
pre-compression (Advanced ACBGEN) 365
PRESPACE keyword (MFS Compare) 427
PRN keyword (MFS Compare) 425
problems
diagnostic information about 522
PROCLIB DD
Advanced ACBGEN 355
Catalog Manager 338
program functions
overview 8
programming interface information 537
progress monitor requested (Advanced ACBGEN) 365
PROMPT keyword (MFS Compare) 423
PSB
library information (DBD/PSB/ACB Reversal) 302
map (DBD/PSB/ACB Mapper) 261
report (DBD/PSB/ACB Mapper) 268
summary report (DBD/PSB/ACB Mapper) 265
PSB Check report (Consistency Checker) 148
PSB Compare report (DBD/PSB/ACB Compare) 227
PSB keyword (Multiple Resource Checker) 175
PSB map (DBD/PSB/ACB Mapper) 261
PSB report (DBD/PSB/ACB Mapper) 268
PSB Size Summary report (Advanced ACBGEN) 368
PSB Summary report (DBD/PSB/ACB Mapper) 265
PSB XREF by Type report
PSB XREF by Type - PSB Name Order
(DBD/PSB/ACB Reversal) 310
PSB XREF by Type - PSB Name Order
(DBD/PSB/ACB Reversal) 310
PSB/DBD Change Summary report
(Advanced ACBGEN) 369
PSBLIB DD
Consistency Checker 131
DBD/PSB/ACB Compare 204
DBD/PSB/ACB Mapper 248
DBD/PSB/ACB Reversal 285
PSBLIB2 DD (DBD/PSB/ACB
Compare) 204
PSBLIBxx DD (Multiple Resource
Checker) 172
PSBSRC data set (DBD/PSB/ACB
Reversal) 308
PSBSRC DD (DBD/PSB/ACB
Reversal) 285

R
RCD keyword (MFS Compare) 425
RCD statement (MFS Reversal) 416
RCDCTL (MFS Reversal) 416
RCDTL keyword (MFS Compare) 425
RDES
creating 56
RDPLN keyword (MFS Compare) 425
reader comment form 21
RECLD keyword (LICON utility) 104
RECON Difference report (Multiple
Resource Checker) 194
RECONx DD
Catalog Manager 338
Consistency Checker 131
LICON utility 98
RECONxx DD (Multiple Resource
Checker) 172
RECOVER.DB command (LICON
utility) 115
RECUL keyword (LICON utility) 104
RECPF keyword (LICON utility) 104
reports
ACB (DBD) report (DBD/PSB/ACB
Mapper) 268
ACB (PSB) report (DBD/PSB/ACB
Mapper) 268
ACB (PSB) Summary report
(DBD/PSB/ACB Mapper) 265
ACB Compare report (DBD/PSB/ACB
Compare) 227
ACB Library Members Deleted Due to
User Request report (Advanced
ACBGEN) 367
ACB/PSB/DBD Library Information
report (Advanced ACBGEN) 365
ACB(DBD) to ACB(DBD) XREF report
(DBD/PSB/ACB Reversal) 315
ACB(DBD) XREF by Access report
(DBD/PSB/ACB Reversal) 309
ACB(DBD) XREF by DName report
(DBD/PSB/ACB Reversal) 312
ACB(PSB) to ACB(DBD) XREF report
(DBD/PSB/ACB Reversal) 317
reports (continued)
ACB(PSB) XREF by Type report
(DBD/PSB/ACB Reversal) 310
ACBLIB Analyzer 386
Advanced ACBGEN 363
Chronological History of ACBGENs
report (ACBLIB Analyzer) 393
DBD Check report (Consistency
Checker) 141
DBD Compare report
(DBD/PSB/ACB Compare) 227
DBD report (DBD/PSB/ACB
Mapper) 268
DBD Segment Reference report
(DBD/PSB/ACB Reversal) 319
DBD to DBD XREF report
(DBD/PSB/ACB Reversal) 315
DBD XREF by Access report
(DBD/PSB/ACB Reversal) 309
DBD XREF by DName report
(DBD/PSB/ACB Reversal) 312
DBD/PSB Names Specified via SYSIN
report (Advanced ACBGEN) 366
DFS Messages Summary report
(Advanced ACBGEN) 372
Distribution of Member Sizes report
(ACBLIB Analyzer) 392
Distribution of PSB Workarea Sizes
report (ACBLIB Analyzer) 393
Final PSB Build List report (Advanced
ACBGEN) 367
Input Specifications report (ACBLIB
Analyzer) 386
Input Specifications report (Advanced
ACBGEN) 363
Library Contents report (ACBLIB
Analyzer) 388
Library Information report (ACBLIB
Analyzer) 387
Load Module Management Status
(Load Module Mgmt Stats) report
(Advanced ACBGEN) 376
Miscellaneous DFS Messages report
(Advanced ACBGEN) 371
PCB PROCOPT report
(DBD/PSB/ACB Reversal) 314
PCB(ACB/PSB) PROCOPT report
(DBD/PSB/ACB Reversal) 314
PSB Check report (Consistency
Checker) 148
PSB Compare report (DBD/PSB/ACB
Compare) 227
PSB PROCOPT reference reports for
PSB and ACB(PSB) (DBD/PSB/ACB
Reversal) 322
PSB report (DBD/PSB/ACB
Mapper) 268
PSB Segment Reference report
(DBD/PSB/ACB Reversal) 321
PSB Size Summary report (Advanced
ACBGEN) 368
PSB Summary report (DBD/PSB/ACB
Mapper) 265
PS to DBD XREF report
(DBD/PSB/ACB Reversal) 317
PSB XREF by Type report
(DBD/PSB/ACB Reversal) 310
reports (continued)
PSB/DBD Change Summary report
(Advanced ACBGEN) 369
RECON Difference report 194
Resource Check Summary report
(Multiple Resource Checker) 190
Run Summary report (Advanced
ACBGEN) 373
Unreferenced ACB(DBD) report
(DBD/PSB/ACB Reversal) 323
Warning Messages report (ACBLIB
Analyzer) 394
REPORTS= operand (Advanced
ACBGEN) 359
Resource Check Summary report
(Multiple Resource Checker) 190
return codes 438
RPRN keyword (MFS Compare) 425
Run Summary report (Advanced
ACBGEN) 373
running
ACBLIB Analyzer 381
Advanced ACBGEN 351
Catalog Manager 337
Consistency Checker 130
DBD/PSB/ACB Compare 203
DBD/PSB/ACB Mapper 247
DBD/PSB/ACB Reversal 284
MFS Compare 428
MFS Reversal 400
Multiple Resource Checker 160
runtime parameters (Advanced
ACBGEN) 365

S
sample generated source statements (MFS
Reversal) 410
sample library 532
sample MFS Compare report 433
SCA keyword (MFS Compare) 423
screen readers and magnifiers 23
SCROLLI keyword (MFS Compare) 427
SCSI devices (MFS Reversal) 417
segment information of DBD
(DBD/PSB/ACB Mapper) 274
SELECT control statement
MFS Compare 431
MFS Reversal 403
SELECT keyword (MFS Compare) 425
service information 20
SHPSJCL0 532
SHPSAMP 532
single-step verification 80, 92
SLDI (MFS Reversal) 417
SLDI keyword (MFS Compare) 424, 426
SLDP keyword (MFS Compare) 424, 426
SNAP= operand (ACBLIB Analyzer) 385
software prerequisites 26
SOR keyword (MFS Compare) 423
SORT DD (ACBLIB Analyzer) 382
source generated by utilities (MFS
Reversal) 398
SPACE option (MFS Reversal) 417
STEPLIB DD
ACBLIB Analyzer 382
Advanced ACBGEN 355
Index 547