

IBM IMS Library Integrity Utilities for z/OS  
2.2

*User's Guide*



**Note:**

Before using this information and the product it supports, read the information in [“Notices” on page 533.](#)

**14th Edition (April 2024)**

This edition applies to Version 2.2 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 SC19-3979-12.

© **Copyright International Business Machines Corporation 2003, 2024.**

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

---

# Contents

<b>About this information.....</b>	<b>ix</b>
<b>Chapter 1. IMS Library Integrity Utilities overview.....</b>	<b>1</b>
What's new in IMS Library Integrity Utilities.....	1
IMS Library Integrity Utilities terminology.....	10
What does IMS Library Integrity Utilities do?.....	11
IMS Library Integrity Utilities solutions.....	15
Functional enhancements in IMS Library Integrity Utilities.....	17
Functional enhancements in IMS Library Integrity Utilities 2.2.....	17
Functional enhancements in IMS Library Integrity Utilities 2.1.....	20
Service updates and support information.....	21
Product documentation and updates.....	21
Accessibility features for IMS Library Integrity Utilities.....	23
<b>Chapter 2. Configuring IMS Library Integrity Utilities.....</b>	<b>25</b>
Hardware and software prerequisites.....	25
Configuring for initial installation.....	26
Setting up security for Consistency Checker and Multiple Resource Checker.....	26
Migration procedures.....	26
Migrating Integrity Checker.....	27
Migrating Advanced ACBGEN.....	31
<b>Chapter 3. Integrity Checker utility.....</b>	<b>33</b>
Integrity Checker overview.....	33
Planning for Integrity Checker configuration.....	36
LIU load module library customization.....	36
LICON data sets and global option modules.....	37
Integrity Checker configuration requirements.....	38
Runtime options and environments.....	43
Historical data maintained in LICON data sets.....	44
Considerations for activating Integrity Checker.....	46
Activating Integrity Checker.....	48
Setting up the global option modules.....	48
Setting up the LICON data sets.....	49
Setting up RACF security.....	51
Customizing LIU load modules.....	53
Configuring for a BPE-based DBRC environment.....	57
Verifying that Integrity Checker is activated.....	58
Restarting IMS online and running IMS batch application, IMS utility, and IMS Tools jobs.....	58
Maintaining Integrity Checker.....	59
Maintaining RDEs.....	59
Maintaining global option modules.....	67
Maintaining LICON data sets.....	69
Restarting Integrity Checker after an abend.....	69
Applying PTFs to IMS Library Integrity Utilities and to IMS.....	69
Preventing database corruption with Integrity Checker .....	70
Restrictions: Cases where DMB verification is not done.....	70
DMB mismatch in IMS online environment or application jobs.....	71
DMB mismatch during database maintenance and operation.....	74
Addressing a DMB mismatch.....	75

Deactivating Integrity Checker.....	76
Deactivating Integrity Checker when IMS Library Integrity Checker is installed as a stand-alone product.....	77
Deactivating Integrity Checker when IMS Library Integrity Utilities is installed as a component of an IMS tools solution pack.....	77
Output from Integrity Checker.....	78
FABLPRNT data set.....	78
FABLSNAP data set.....	79
Global option module generation macro.....	79
Creating global option modules.....	79
JCL requirements for the FABLPGEN program.....	80
LICON utility reference.....	85
JCL requirements for the LICON utility.....	86
Input for the LICON utility.....	88
Runtime options.....	88
INIT.DB command.....	89
INIT.LICON command.....	95
CHANGE.DB command.....	95
DELETE.DB command.....	100
EXPIRE.DB command.....	101
LIST.DB command.....	102
LIST.LICON command.....	103
RECOVER.DB command.....	104
VERIFY.DB command.....	105
Output from the LICON utility.....	106

#### **Chapter 4. Consistency Checker utility..... 113**

Consistency Checker utility overview.....	113
Restriction for Consistency Checker.....	115
Checking the consistency of definitions .....	115
JCL requirements for the Consistency Checker utility.....	116
Control statements for the Consistency Checker utility.....	118
JCL examples for the Consistency Checker utility.....	120
Example: Checking the consistency of DBDs.....	120
Example: Checking the consistency of PSBs.....	120
Example: Checking the consistency of DBDs and PSBs.....	121
Output from the Consistency Checker utility.....	121
SYSOUT data set.....	121
SYSPRINT data set.....	122

#### **Chapter 5. Multiple Resource Checker utility..... 135**

Multiple Resource Checker utility overview.....	135
Checking consistencies with the Multiple Resource Checker utility.....	136
Checking the consistency of multiple resources.....	137
Checking the consistency of multiple sets of RECON data sets.....	139
Checking the consistency across two resource types .....	142
Checking the consistency of same resource-type members in multiple libraries.....	144
JCL requirements for the Multiple Resource Checker utility.....	146
Control statements for the Multiple Resource Checker utility.....	148
Fields compared in RECON data sets.....	150
JCL examples for the Multiple Resource Checker.....	155
Examples: Checking the consistency of multiple resources.....	155
Example: Comparing the database definitions across multiple sets of RECON data sets.....	158
Output from the Multiple Resource Checker utility.....	159
FABWOUT data set.....	159
FABWSUMM data set.....	159
FABWRRPT data set.....	162

<b>Chapter 6. DBD/PSB/ACB Compare utility.....</b>	<b>167</b>
DBD/PSB/ACB Compare utility overview.....	167
Restrictions and considerations for the DBD/PSB/ACB Compare utility.....	168
Comparing IMS control blocks.....	169
JCL requirements for the DBD/PSB/ACB Compare utility.....	170
Control statements for the DBD/PSB/ACB Compare utility.....	171
DBD, PSB, ACB control statements.....	172
REPORT control statement.....	174
NOCOMP control statement.....	175
JCL examples for the DBD/PSB/ACB Compare utility.....	184
Example: Comparing two DBDs.....	184
Example: Comparing two DBDs that have different names.....	184
Example: Comparing a DBD with a DBD-type ACB.....	185
Example: Comparing two PSBs.....	185
Example: Comparing a PSB with a PSB-type ACB.....	185
Example: Comparing two ACBs.....	186
Example: Comparing ACBs with DBDs and PSBs.....	186
Example: Comparing DBDs, PSBs, and ACBs.....	187
Output from the DBD/PSB/ACB Compare utility.....	187
SYSOUT data set.....	187
SYSPRINT data set.....	187
<b>Chapter 7. DBD/PSB/ACB Mapper utility.....</b>	<b>201</b>
DBD/PSB/ACB Mapper utility overview.....	201
Restrictions for the DBD/PSB/ACB Mapper utility.....	202
Printing hierarchical structure of databases.....	203
JCL requirements for the DBD/PSB/ACB Mapper utility.....	203
Control statements for the DBD/PSB/ACB Mapper utility.....	204
JCL examples for the DBD/PSB/ACB Mapper utility.....	206
Example: Generating DBD maps.....	206
Example: Generating PSB maps.....	206
Example: Generating ACB maps.....	207
Example: Generating DBD, PSB, ACB maps.....	208
Example: Creating a DBD and generating a DBD map.....	208
Example: Creating a PSB and generating a PSB map.....	208
Output from the DBD/PSB/ACB Mapper utility.....	209
SYSOUT data set.....	209
SYSPRINT data set.....	210
<b>Chapter 8. DBD/PSB/ACB Reversal utility.....</b>	<b>225</b>
DBD/PSB/ACB Reversal utility overview.....	225
Restrictions for the DBD/PSB/ACB Reversal utility.....	227
Converting IMS control blocks to control statements.....	228
JCL requirements for the DBD/PSB/ACB Reversal utility.....	229
Control statements for the DBD/PSB/ACB Reversal utility.....	231
JCL examples for the DBD/PSB/ACB Reversal utility.....	241
Example: Re-creating the sources from DBDs and PSBs.....	241
Example: Re-creating the sources from ACBs.....	241
Example: Obtaining DBD library information.....	242
Example: Obtaining PSB library information.....	242
Example: Obtaining control statement source and Mapper input.....	243
Output from the DBD/PSB/ACB Reversal utility.....	243
SYSOUT data set.....	243
SYSPUNCH data set.....	244
DBDSRC data set.....	247
PSBSRC data set.....	247

SYSPRINT data set.....	247
MAPOUT data set.....	261
OPTPRT data set.....	261
DBD/PSB/ACB Reversal Site Default Generation utility.....	262
Reversal Site Default Generation utility overview.....	262
Setting site default values for the DBD/PSB/ACB Reversal utility.....	262
JCL requirements for the Reversal Site Default Generation utility.....	264
Control statements for the Reversal Site Default Generation utility.....	266
Output from the DBD/PSB/ACB Reversal Site Default Generation utility.....	266
<b>Chapter 9. MDA Reversal utility.....</b>	<b>269</b>
MDA Reversal utility overview.....	269
MDA Reversal utility restrictions.....	270
Converting DFSMDA members back into DFSMDA macros.....	271
JCL requirements for the MDA Reversal utility.....	271
Control statements for the MDA Reversal utility.....	272
JCL examples for the MDA Reversal utility.....	275
Output from the MDA Reversal utility.....	275
FABXMSRC data set.....	275
MDASRC data set.....	278
FABXMOUT data set.....	278
FABXMRPT data set.....	279
<b>Chapter 10. Catalog Manager utility.....</b>	<b>281</b>
Catalog Manager utility overview.....	281
Catalog Manager utility restrictions.....	284
Validating IMS control blocks in the IMS catalog.....	286
Comparing IMS control blocks .....	286
Converting IMS control blocks to control statements.....	287
Generating maps of IMS control blocks in the IMS catalog.....	288
JCL requirements for the Catalog Manager utility.....	288
Control statements for the Catalog Manager utility.....	292
Control statements for the validate function.....	293
Control statements for the compare function.....	294
Control statements for the convert function.....	298
Control statements for the map function.....	301
JCL examples for the Catalog Manager utility.....	304
Example: Validating DBDs and PSBs.....	304
Example: Comparing IMS control blocks.....	304
Example: Converting IMS control blocks to control statements.....	308
Example: Generating maps and reports.....	308
Output from the Catalog Manager utility.....	311
Output from the validate function.....	311
Output from the compare function.....	317
Output from the convert function.....	321
Output from the map function.....	324
<b>Chapter 11. Advanced Application Control Block Generator utility.....</b>	<b>333</b>
Advanced ACBGEN utility overview.....	333
Generating application control blocks.....	334
Merging Advanced ACBGEN load modules into the IMS SDFSRESL library.....	334
Using the Advanced ACBGEN utility in an ACB Generation and Catalog Populate utility job.....	335
JCL requirements for the Advanced ACBGEN utility.....	336
Control statements for the Advanced ACBGEN utility.....	338
SYSIN control statements.....	339
ACBSYSIN control statements.....	339
Output from the Advanced ACBGEN utility.....	342

SYSPRINT data set.....	342
DFSPRINT data set.....	353
MVS console and the JES job listing.....	354
<b>Chapter 12. ACBLIB Analyzer utility.....</b>	<b>355</b>
ACBLIB Analyzer utility overview.....	355
Analyzing ACB libraries.....	355
JCL requirements for the ACBLIB Analyzer utility.....	355
ACBSYSIN control statements.....	357
Output from the ACBLIB Analyzer utility.....	358
Input Specifications report.....	359
Library Information report.....	359
Library Contents report.....	360
Distribution of Member Sizes report.....	363
Distribution of PSB Workarea Sizes report.....	364
Chronological History of ACBGENs report.....	364
Warning Messages report.....	365
<b>Chapter 13. MFS Reversal utility.....</b>	<b>367</b>
MFS Reversal utility overview.....	367
Restrictions and considerations for the MFS Reversal utility.....	368
Converting MFS control blocks to control statements.....	368
JCL requirements for the MFS Reversal utility.....	368
Control statements for the MFS Reversal utility.....	370
Output from the MFS Reversal utility.....	373
SYSOUT data set.....	373
SYSPRINT data set.....	373
MFSSRCE data set.....	375
COPYFMT data set.....	377
COPYPRT data set.....	377
Important notes about the generated source.....	379
<b>Chapter 14. MFS Compare utility.....</b>	<b>381</b>
MFS Compare utility overview.....	381
Considerations for the MFS Compare utility.....	382
Keywords used in comparisons.....	382
Comparing MFS control blocks.....	386
JCL requirements for the MFS Compare utility.....	387
Control statements for the MFS Compare utility.....	389
Output from the MFS Compare utility.....	390
SYSOUT data set.....	390
SYSPRINT data set.....	390
<b>Chapter 15. Troubleshooting.....</b>	<b>393</b>
IMS Library Integrity Utilities return codes .....	393
Integrity Checker and LICON utility return codes.....	393
Consistency Checker return codes.....	394
Multiple Resource Checker return codes.....	394
DBD/PSB/ACB Compare, Mapper, and Reversal return codes.....	395
MDA Reversal return codes.....	395
Catalog Manager return codes.....	396
Advanced ACB Generator return codes.....	396
MFS Reversal return codes.....	396
MFS Compare return codes.....	397
IMS Library Integrity Utilities return codes under IMS Administration Tool.....	397
IMS Library Integrity Utilities abend codes.....	399
Integrity Checker abend codes.....	399

Consistency Checker abend codes.....	399
Multiple Resource Checker abend codes.....	399
DBD/PSB/ACB Compare abend codes.....	399
DBD/PSB/ACB Mapper abend codes.....	399
DBD/PSB/ACB Reversal abend codes.....	399
MFS Reversal abend codes.....	400
IMS Library Integrity Utilities abend codes when run under other tools to write DBD and PSB maps .....	400
IMS messages.....	400
IMS Library Integrity Utilities messages .....	400
FABL messages.....	401
FABM messages.....	446
FABN messages.....	453
FABQ messages.....	472
FABV messages.....	487
FABW messages.....	495
FABX messages.....	499
How to look up message explanations.....	519
Gathering diagnostic information.....	520
Diagnostics Aid.....	520
How to run Diagnostics Aid with JCL.....	520
Load Module/Macro APAR Status report.....	521
Diagnostic Aid messages and codes.....	522
<b>Chapter 16. References.....</b>	<b>527</b>
Device and feature code tables.....	527
Sample library members.....	529
How to read syntax diagrams.....	530
<b>Notices.....</b>	<b>533</b>
<b>Index.....</b>	<b>537</b>



## 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:

<https://www.ibm.com/support/pages/node/712955>

The IMS Tools Product Documentation web page includes:

- Links to [IBM Documentation](#) for the user guides ("HTML")
- PDF versions of the user guides ("PDF")
- Program Directories for IMS Tools products
- Technical notes from IBM Software Support, referred to as "Tech notes"
- White papers that describe product business scenarios and solutions



---

# Chapter 1. IMS Library Integrity Utilities overview

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 DBD libraries, PSB libraries, ACB libraries, RECON data sets, IMS catalog, IMS directory, and libraries containing DFSMDA members that you use when referring to IMS databases.

## Topics:

- [“What's new in IMS Library Integrity Utilities” on page 1](#)
- [“IMS Library Integrity Utilities terminology” on page 10](#)
- [“What does IMS Library Integrity Utilities do?” on page 11](#)
- [“IMS Library Integrity Utilities solutions” on page 15](#)
- [“Functional enhancements in IMS Library Integrity Utilities” on page 17](#)
- [“Service updates and support information” on page 21](#)
- [“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.

Revision markers follow these general conventions:

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

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

### SC19-3979-13 (April 2024)

Description	Related APARs
New messages FABX0570E, FABX0571E, and FABX0572I have been added for IMS Library Integrity Utilities that is run under IMS Administration Tool.	PH60138

### SC19-3979-12 (January 2024)

Description	Related APARs
New messages FABL0055W, FABN0087W, FABN0091E, and FABX0014W have been added and messages FABL0029E and FABN0036E have been modified for the Catalog Manager utility and IMS Library Integrity Utilities that is run under IMS Administration Tool.	PH58376
Explanation of return code 4 for the Catalog Manager utility has also been updated.	

Description	Related APARs
Removed information related to IBM Management Console for IMS and Db2® for z/OS.	N/A

### SC19-3979-11 (July 2023)

Description	Related APARs
<p>Information about abend codes when IMS Library Integrity Utilities is run under other tools has been added. For more information, see <a href="#">“IMS Library Integrity Utilities abend codes when run under other tools to write DBD and PSB maps ” on page 400.</a></p> <p>Also, a new message FABX0569W has been added to support IMS Administration Foundation.</p>	PH54565

### SC19-3979-10 (April 2023)

Description	Related APARs
<p>This APAR enables the Catalog Manager utility to provide the map function of the IMS catalog database. For more information, see the following topics:</p> <ul style="list-style-type: none"> <li>• <a href="#">“What does IMS Library Integrity Utilities do?” on page 11</a></li> <li>• <a href="#">“IMS Library Integrity Utilities solutions” on page 15</a></li> <li>• <a href="#">“Functional enhancements in IMS Library Integrity Utilities 2.2” on page 17</a></li> <li>• <a href="#">“Catalog Manager utility overview” on page 281</a></li> <li>• <a href="#">“Catalog Manager utility restrictions” on page 284</a></li> <li>• <a href="#">“Generating maps of IMS control blocks in the IMS catalog” on page 288</a></li> <li>• <a href="#">“JCL requirements for the Catalog Manager utility” on page 288</a></li> <li>• <a href="#">“Control statements for the map function” on page 301</a></li> <li>• <a href="#">“Example: Generating maps and reports” on page 308</a></li> <li>• <a href="#">“Output from the map function” on page 324</a></li> <li>• New messages FABX0565E, FABX0566I, FABX0567W, and FABX0568W</li> </ul>	PH52559

## SC19-3979-09 (August 2022)

Description	Related APARs
<p>This APAR enables the following LIU utilities to support DEDB which has AREAs over 2048 with IMS APAR PH12671.</p> <ul style="list-style-type: none"><li>• Integrity Checker utility</li><li>• Consistency Checker utility</li><li>• DBD/PSB/ACB Compare utility</li><li>• DBD/PSB/ACB Mapper utility</li><li>• DBD/PSB/ACB Reversal utility</li><li>• Catalog Manager utility</li><li>• Advanced ACBGEN utility</li><li>• ACBLIB Analyzer utility</li><li>• LIU under the IMS Administration Tool</li></ul> <p>The explanation about the maximum number of AREAs in <a href="#">“DBD, PSB, and ACB maps”</a> on page 210 has been updated.</p> <p>Also, layout of example reports in <a href="#">“Library Contents report”</a> on page 360 has been updated.</p>	PH48231

## SC19-3979-08 (July 2022)

Description	Related APARs
<p>Documentation updates to support IMS Administration Foundation, which activates the IMS administration web-browser interface of IBM Unified Management Server for z/OS to enable the management of IMS systems and resources. For more information, see the following topics:</p> <ul style="list-style-type: none"><li>• <a href="#">“Functional enhancements in IMS Library Integrity Utilities 2.2”</a> on page 17</li><li>• <a href="#">“Checking the consistency of multiple resources”</a> on page 137</li><li>• <a href="#">“Checking the consistency across two resource types ”</a> on page 142</li><li>• <a href="#">“Checking the consistency of same resource-type members in multiple libraries”</a> on page 144</li><li>• Modified messages FABX3902E and FABX3904E</li></ul>	N/A
<p><b>DBD/PSB Map Viewer</b></p> <p>Changes in prerequisite software for the DBD/PSB Map Viewer. Formerly, IMS Database Solution Pack, IMS Database Utility Solution, or IMS Fast Path Solution Pack was required. After the PTF of this APAR is applied, the DBD/PSB Map Viewer can be activated without IMS solution packs.</p>	PH47086

## SC19-3979-07 (April 2022)

Description	Related APARs
<p><b>Consistency Checker utility</b></p> <p>Documentation changes related to reporting the IMS installed level. For more information, see the following topics:</p> <ul style="list-style-type: none"><li>• <a href="#">“DBD Check report”</a> on page 122</li><li>• <a href="#">“PSB Check report”</a> on page 128</li></ul>	PH44974

Description	Related APARs
<b>Catalog Manager utility</b> Documentation changes related to restrictions for the Catalog Manager utility. For more information, see <a href="#">“Catalog Manager utility restrictions” on page 284.</a>	PH43839
Documentation changes related to reporting the IMS installed level. For more information, see the following topics: <ul style="list-style-type: none"> <li>• <a href="#">“FABXCRP1 data set (Validate function)” on page 312</a></li> <li>• <a href="#">“FABXCRP1 data set (Compare function)” on page 317</a></li> <li>• <a href="#">“FABXCRP1 data set (Convert function)” on page 321</a></li> </ul>	PH44974
Notes have been added to explain the IMS level shown in the reports. For more information, see the following topics: <ul style="list-style-type: none"> <li>• <a href="#">“Source-level compare reports” on page 195</a></li> <li>• <a href="#">“SYSPRINT data set” on page 210</a></li> <li>• <a href="#">“SYSPUNCH data set” on page 244</a></li> <li>• <a href="#">“Library member list report for DBD or PSB” on page 260</a></li> <li>• <a href="#">“Input Specifications report” on page 342</a></li> <li>• <a href="#">“Library Contents report” on page 360</a></li> </ul>	N/A

## SC19-3979-06 (February 2022)

Description	Related APARs
<p><b>Integrity Checker utility</b></p> <p>The LICON utility has been enhanced to use IMS directory as input. For more information, see the following topics:</p> <ul style="list-style-type: none"><li>• <a href="#">“Integrity Checker overview” on page 33</a></li><li>• <a href="#">“LICON data sets and global option modules” on page 37</a></li><li>• <a href="#">“Activating Integrity Checker” on page 48</a></li><li>• <a href="#">“Creating an RDE to register DMB information” on page 50</a></li><li>• <a href="#">“Maintaining Integrity Checker” on page 59</a></li><li>• <a href="#">“RDE maintenance at database reorganization” on page 60</a></li><li>• <a href="#">“RDE maintenance at DBD or RECON change” on page 63</a></li><li>• <a href="#">“DMB mismatch in IMS online environment or application jobs” on page 71</a></li><li>• <a href="#">“DMB mismatch during database maintenance and operation” on page 74</a></li><li>• <a href="#">“Addressing a DMB mismatch” on page 75</a></li><li>• <a href="#">“LICON utility reference” on page 85</a></li><li>• <a href="#">“JCL requirements for the LICON utility” on page 86</a></li><li>• <a href="#">“Input for the LICON utility” on page 88</a></li><li>• <a href="#">“Runtime options” on page 88</a></li><li>• <a href="#">“INIT.DB command” on page 89</a></li><li>• <a href="#">“VERIFY.DB command” on page 105</a></li><li>• New messages <a href="#">“FABL0432E” on page 422</a>, <a href="#">“FABL0433E” on page 422</a>, <a href="#">“FABL3416E” on page 445</a>, <a href="#">“FABL3417E” on page 445</a>, and <a href="#">“FABL3418E” on page 445</a></li><li>• Modified message <a href="#">“FABL0453I” on page 424</a></li></ul>	PH24779
<p><b>DBD/PSB/ACB Compare utility</b></p> <p>PXXREF DB section has been added to the PSB Compare reports. For more information, see <a href="#">“Block-level compare reports” on page 188</a>.</p>	PH15721
<p>A new keyword, NOREFPSB, has been added to the REPORT statement to specify not to refer to a PSB-type ACB when the utility decodes an ACB of DEDB or MSDB to be compared in source-level comparison. For more information, see <a href="#">“REPORT control statement” on page 174</a>.</p>	PH38582
<p><b>DBD/PSB/ACB Reversal utility</b></p> <p>A new keyword, REFER_PSB, has been added to specify whether to refer to a PSB-type ACB when the utility decodes an ACB for DEDB or MSDB. For more information, see <a href="#">“Control statements for the DBD/PSB/ACB Reversal utility” on page 231</a>.</p>	PH34243
<p>A new keyword, DECOPT FORMAT_COL10, has been added to specify whether to print the decoded DBDGEN or PSBGEN macro statements in the new format. For more information, see the following topics:</p> <ul style="list-style-type: none"><li>• <a href="#">“Control statements for the DBD/PSB/ACB Reversal utility” on page 231</a></li><li>• <a href="#">“Control statements for the Reversal Site Default Generation utility” on page 266</a></li></ul>	PH40497

Description	Related APARs
<p><b>Catalog Manager utility</b></p> <p>A new keyword, REFER_PSB, has been added to specify whether to refer to a PSB-type ACB when the utility decodes an ACB for DEDB or MSDB. For more information, see the following topics:</p> <ul style="list-style-type: none"> <li>• <a href="#">“Control statements for the convert function” on page 298</a></li> <li>• <a href="#">“FABXCRPO data set (Convert function)” on page 321</a></li> </ul>	PH34243
<p>A new keyword, REFER_PSB, has been added to specify whether to refer to a PSB-type ACB when the utility decodes an ACB for DEDB or MSDB to be compared. For more information, see the following topics:</p> <ul style="list-style-type: none"> <li>• <a href="#">“Control statements for the compare function” on page 294</a></li> <li>• <a href="#">“FABXCRPO data set (Compare function)” on page 317</a></li> </ul>	PH38582
<p>A new keyword, FORMAT_COL10, has been added to specify whether to print the decoded DBDGEN or PSBGEN macro statements in the new format. For more information, see the following topics:</p> <ul style="list-style-type: none"> <li>• <a href="#">“Control statements for the convert function” on page 298</a></li> <li>• <a href="#">“FABXCRPO data set (Convert function)” on page 321</a></li> </ul>	PH40497
<p><b>Troubleshooting</b></p> <p>Messages <a href="#">“FABL0053W” on page 411</a>, <a href="#">“FABN0090W” on page 468</a>, <a href="#">“FABX0559W” on page 511</a>, <a href="#">“FABX0560W” on page 511</a>, and <a href="#">“FABX0561W” on page 511</a> have been added.</p>	PH18802
<p>Message <a href="#">“FABX0562E” on page 511</a> has been added, and <a href="#">“FABX0546E” on page 509</a> has been modified.</p>	PH30836
<p>Message <a href="#">“FABX0011W” on page 501</a> has been modified.</p>	PH34243
<p>Message <a href="#">“FABX0563E” on page 512</a> has been added.</p>	PH37293
<p>Message <a href="#">“FABL0054W” on page 411</a> has been added.</p>	PH38582

## SC19-3979-05 (August 2019)

### Catalog Manager utility

- The utility has been enhanced to support BMP regions (APAR PH04670). For more information, see [“DD statements” on page 289](#).
- The function to validate IMS control blocks in the IMS catalog and the IMS directory has been enhanced to support the IMS directory staging data set (APAR PH09134). By this enhancement, the algorithm for validating the time stamps of ACBs in the ACB libraries and in the IMS directory and instances in the IMS catalog database has changed. Also, the format of the IMS Catalog Validation report has been changed.

For more information, see the following topics:

- [“Catalog Manager utility overview” on page 281](#)
- [“Output from the validate function” on page 311](#)



## SC19-3979-04 (October 2018)

### MDA Reversal utility

A new utility, MDA Reversal, is added. The utility converts DFSMDA members to DFSMDA macros (APAR PI98748). The utility can also generate a report that lists information about all the DFSMDA members found in libraries that you specify. For more information, see [Chapter 9, “MDA Reversal utility,” on page 269.](#)

### Catalog Manager utility

The utility supports the following new functions (APAR PI95272).

- Compare IMS control blocks in the IMS directory with those in ACB libraries, DBD libraries, or PSB libraries.
- Convert ACBs in the IMS directory into IMS DBDGEN control statements or into IMS PSBGEN control statements.

The utility has also been enhanced to compare IMS control blocks in the IMS directory (APAR PH00141).

For more information, see [Chapter 10, “Catalog Manager utility,” on page 281.](#)

## 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 175.](#)
- The utility supports new parameters for the NOCOMP statement (APAR PI71280). The new parameters are LANG, LIST, PROCOPT, PROCSEQ, PROCSEQD, PSB\_ACCESS, and PSB\_PSELOPT. For more information, see [“NOCOMP control statement” on page 175.](#)

### 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 15, “Troubleshooting,” on page 393.](#)

## 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 171.](#)

### 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 210.](#)

### 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 231](#)
- [“Unreferenced ACB\(DBD\) report” on page 259](#)
- 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 231](#)
  - [“Library member list report for DBD or PSB” on page 260](#)

### **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 281](#).

### **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 46](#)
- [“Altering the definition of a DMB verification-enabled online HALDB by using the HALDB alter function” on page 63](#)
- [“Altering the definition of a DMB verification-enabled online DEDB by using the DEDB Alter utility \(DBFUDA00\)” on page 65](#)

### **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 135](#)
- [“Checking the consistency of multiple sets of RECON data sets” on page 139](#)
- [“JCL requirements for the Multiple Resource Checker utility” on page 146](#)
- [“Control statements for the Multiple Resource Checker utility” on page 148](#)
- [“Fields compared in RECON data sets” on page 150](#)
- [“FABWSUMM data set” on page 159](#)
- [“FABWRRPT data set” on page 162](#)

### **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 168](#)
  - [“Control statements for the DBD/PSB/ACB Compare utility” on page 171](#)
- 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 167](#)
  - [“Restrictions and considerations for the DBD/PSB/ACB Compare utility” on page 168](#)
  - [“Control statements for the DBD/PSB/ACB Compare utility” on page 171](#)

### **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 blockThe report examples were updated in [“SYSPUNCH data set” on page 244](#).
- 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 225](#)
  - [“Control statements for the DBD/PSB/ACB Reversal utility” on page 231](#)
  - [“PSB PROCOPT reference reports for PSB and ACB\(PSB\)” on page 258](#)

### **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 10, “Catalog Manager utility,” on page 281](#).

### **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 355](#)
- [“ACBSYSIN control statements” on page 357](#)
- [“Output from the ACBLIB Analyzer utility” on page 358](#)
- [“Library Contents report” on page 360](#)

Also see [“Functional enhancements in IMS Library Integrity Utilities 2.2” on page 17](#) 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 36](#).

### **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 135](#).

### 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 167.

### Others

Supports the DBD Map Viewer, which is the IMS Library Integrity Utilities extension for Management 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 IMS Library Integrity Utilities 2.2”](#) on page 17.

## 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 14* refers to IMS 14.1 and later, and IMS Database Value Unit Edition 14.1 and later.
- *IMS 15* refers to IMS 15.1 and later, and IMS Database Value Unit Edition 15.1 and later.

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*

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

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

Table 2. Product short names

Short name	Product name
IMS Administration Foundation	IMS Administration Foundation features activated on top of IBM Unified Management Server for z/OS
IMS Administration Tool	IBM IMS Administration Tool for z/OS
IMS Database Recovery Facility	IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility
IMS Database Reorganization Expert	IBM IMS Database Reorganization Expert for z/OS
IMS HP Fast Path Utilities	IBM IMS Fast Path Solution Pack for z/OS: IMS High Performance Fast Path Utilities
IMS HP Image Copy	IBM IMS High Performance Image Copy for z/OS
IMS HP Load	IBM IMS High Performance Load for z/OS
IMS Library Integrity Utilities or IMS LIU	IBM IMS Library Integrity Utilities for z/OS (this product)
IMS Online Reorganization Facility	IBM IMS Database Solution Pack for z/OS: IMS Online Reorganization Facility

## What does IMS Library Integrity Utilities do?

IMS Library Integrity Utilities aids you in managing data for the libraries, such as DBD libraries, PSB libraries, ACB libraries, RECON data sets, IMS catalog, IMS directory, and libraries containing DFSMDA members 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” on page 12](#)
- [“Consistency Checker utility” on page 12](#)
- [“Multiple Resource Checker utility” on page 13](#)
- [“DBD/PSB/ACB Compare utility” on page 13](#)
- [“DBD/PSB/ACB Mapper utility” on page 13](#)
- [“DBD/PSB/ACB Reversal utility” on page 13](#)
- [“MDA Reversal utility” on page 14](#)
- [“Catalog Manager utility” on page 14](#)
- [“Advanced ACB Generator utility” on page 14](#)
- [“ACBLIB Analyzer utility” on page 15](#)
- [“MFS Reversal utility” on page 15](#)
- [“MFS Compare utility” on page 15](#)

**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 113](#).

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.

## MDA Reversal utility

The MDA Reversal utility converts DFSMDA members back into DFSMDA macros. Also, the utility generates a report that lists DFSMDA members and their properties.

- Convert DFSMDA members back into DFSMDA macros – converts DFSMDA members back into DFSMDA macros. This function reads one or more DFSMDA members from the specified library and converts them back into DFSMDA macros.
- Generate a report that contains a list of DFSMDA members – generates the Library Contents report which contains a list of DFSMDA members in the specified library. Detailed information about DFSMDA members, such as the name of the DD statement and the name of the data set, is shown for each DFSMDA member.

## Catalog Manager utility

The Catalog Manager utility provides the capabilities to ensure that the IMS catalog and the IMS directory are maintained correctly and to analyze DBDs and PSBs in the IMS directory.

- Time stamp validation – checks DBDs and PSBs in the IMS catalog to ensure that they are maintained correctly. It checks the time stamps of DBDs and PSBs in the IMS catalog, the IMS directory, and ACB libraries to ensure that they are consistent.
- Compare IMS control blocks – compares IMS control blocks and reports the differences. The utility supports comparing IMS control blocks (DBD-type ACBs and PSB-type ACBs) within the IMS directory, and those between the IMS directory and ACB libraries. The utility also supports comparing those resources in the IMS directory with DBDs in DBD libraries and PSBs in PSB libraries.
- Convert IMS control blocks to IMS DBDGEN/PSBGEN utility control statements – converts ACBs in the IMS directory back into IMS DBDGEN/PSBGEN utility control statements. This function is useful, for example, when you no longer have the original source for ACBs and you need to re-create them.
- Map IMS control blocks – generates maps from DBDs and PSBs in the IMS catalog database. The utility depicts the hierarchical structure of a database as described in a DBD and reports the DBD and PSB details.

## 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. IMS uses database management blocks (DMB), which are stored in DBD libraries (DBDLIB) or ACB libraries (ACBLIB) as load modules, to obtain database definition when IMS databases are accessed by IMS online applications,

batch applications, utilities, or IMS Tools jobs. If Integrity Checker is activated, it refers to the DMB information that is registered to the LICON data set (Library Integrity Control data set) at the time of database load, and ensures, every time IMS gets access to databases, that the ACB or DBD that is being used is the correct one by comparing the DMB and the DMB information in the LICON data set. Integrity Checker can be activated either online or by batch. If any inconsistency exists, it must be resolved, and to resolve the inconsistency, you can use the following utilities that IMS Library Integrity Utilities provides.

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.
- MDA Reversal converts DFSMDA members to DFSMDA macros.

If the IMS management of ACBs is enabled or you plan to migrate to the environment in which the IMS management of ACBs is enabled, the following utilities can help you.

Catalog Manager, for example, to migrate ACBs from ACB libraries to the IMS management of ACBs environment, or to maintain IMS resources if the IMS management of ACBs is enabled. If the IMS management of ACBs is enabled and you are maintaining IMS control blocks (DBDs and PSBs) by

populating the IMS catalog, you must always ensure that the IMS catalog and the IMS directory are in sync with ACB libraries. You can do so by using Catalog Manager. Catalog Manager has the following capabilities:

- Validating IMS control blocks
- Comparing IMS control blocks
- Converting IMS control blocks to control statements
- Generating maps and reports of IMS control blocks in the IMS catalog database

Advanced ACBGEN can also be used in such an environment. It can replace the IMS ACBGEN utility that is used to generate ACB members within ACB Generation and Catalog Populate utility (DFS3UACB) jobs.

## Functional enhancements in IMS Library Integrity Utilities

---

The major functional enhancements that have been made in IMS Library Integrity Utilities are described in this topic.

### Functional enhancements in IMS Library Integrity Utilities 2.2

The major functional enhancements made to IMS Library Integrity Utilities 2.2 are described in this topic.

Subsections:

- [“Integrity Checker utility” on page 17](#)
- [“Multiple Resource Checker utility” on page 17](#)
- [“DBD/PSB/ACB Compare utility” on page 17](#)
- [“DBD/PSB/ACB Mapper utility” on page 18](#)
- [“DBD/PSB/ACB Reversal utility” on page 18](#)
- [“MDA Reversal utility” on page 18](#)
- [“Catalog Manager utility” on page 19](#)
- [“ACBLIB Analyzer utility” on page 19](#)
- [“Other enhancements” on page 19](#)

#### Integrity Checker utility

APAR PH24779 enhances the LICON utility to use IMS directory as input. For more information, see Chapter 3, [“Integrity Checker utility,” on page 33](#).

#### 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 135](#).

#### 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.
- APAR PH38582 adds a new keyword, NOREFPSB, to the REPORT statement for decoding condition that specifies not to refer to a PSB-type ACB when the utility decodes an ACB of DEDB or MSDB to be compared in source-level comparison.

For more information, see [Chapter 6, “DBD/PSB/ACB Compare utility,” on page 167.](#)

### **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 201.](#)

### **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.
- APAR PH34243 adds a new keyword, REFER\_PSB, for decoding condition that specifies whether to refer to a PSB-type ACB when the utility decodes an ACB for DEDB or MSDB.
- APAR PH40497 adds a new option keyword, DECOPT FORMAT\_COL10, for printing the decoded DBDGEN or PSBGEN macro statements in the new format.

For more information, see [Chapter 8, “DBD/PSB/ACB Reversal utility,” on page 225.](#)

### **MDA Reversal utility**

APAR PI98748 provides a new utility, the MDA Reversal utility. This new utility converts DFSMDA members to DFSMDA macros. The utility can also generate a report that lists information about all the

DFSMDA members found in libraries that you specify. For more information, see [Chapter 9, “MDA Reversal utility,”](#) on page 269.

## 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.
- APAR PI95272 provides the following new functions:
  - Compare IMS control blocks in the IMS directory with those in ACB libraries, DBD libraries, or PSB libraries. You can compare ACBs in the IMS directory with those in ACB libraries, or compare ACBs in the IMS directory with DBDs in DBD libraries or PSBs in PSB libraries.
  - Convert ACBs in the IMS directory into IMS DBDGEN control statements or into IMS PSBGEN control statements.
- APAR PH00141 provides the function to compare IMS control blocks in the IMS directory.
- APAR PH09134 enhances the validate function to support ACBs in the IMS directory staging data set. The validate function can check the time stamps of ACBs in the IMS directory staging data set to ensure the consistency of the ACBs in the IMS directory staging data set.
- APAR PH34243 adds a new keyword, REFER\_PSB, for decoding condition that specifies whether to refer to a PSB-type ACB when the utility decodes an ACB for DEDB or MSDB.
- APAR PH38582 adds a new keyword, REFER\_PSB, for decoding condition that specifies whether to refer to a PSB-type ACB when the utility decodes an ACB for DEDB or MSDB to be compared.
- APAR PH40497 adds a new keyword, FORMAT\_COL10, for printing the decoded DBDGEN or PSBGEN macro statements in the new format.
- APAR PH52559 provides a new function to generate maps and reports of the IMS catalog database. These maps are similar to the maps and reports that the DBD/PSB/ACB Mapper utility generates.

For more information, see [Chapter 10, “Catalog Manager utility,”](#) on page 281.

## 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 360.

## Other enhancements

- Provides the DBD/PSB map feature of IMS Administration Foundation. It displays the graphical visualization of database segment tree structures, program specifications, and DBD and PSB macro source statements. You can also use the cross reference feature to list logical DBDs and PSBs that refer to a specific DBD.

You must configure the environment before you can use this feature. For more information, see the customization topics in the *Overview and Customization* guide of the solution packs.

- IMS Library Integrity Utilities extends the functions of IBM IMS Administration Tool for z/OS (APAR PI67745, PI89811, and PH00141).

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*.

## Functional enhancements in IMS Library Integrity Utilities 2.1

The major functional enhancements made to IMS Library Integrity Utilities 2.1 are described in this topic.

Subsections:

- [“Integrity Checker utility” on page 20](#)
- [“DBD/PSB/ACB Compare utility” on page 20](#)
- [“DBD/PSB/ACB Reversal utility” on page 21](#)
- [“Advanced ACB Generator utility” on page 21](#)
- [“MFS Reversal utility” on page 21](#)

### 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.

#### 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 80](#) and [“INIT.DB command” on page 89](#).

#### 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

#### Support for global option modules prefixed with LIUG

Integrity Checker supports global option modules that are prefixed with LIUG. You can use LIUGINST or LIUG $\textit{imsid}$  in addition to LIU@INST and LIU@ $\textit{imsid}$ . 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 167](#).

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 262.

### **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 333.

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 336.

### **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:

[http://www.ibm.com/support/entry/portal/Overview/Software/Information\\_Management/IMS\\_Tools](http://www.ibm.com/support/entry/portal/Overview/Software/Information_Management/IMS_Tools)

## **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:

<https://www.ibm.com/support/pages/node/712955>

The IMS Tools Product Documentation web page includes:

- Links to [IBM Documentation](#) for the user guides ("HTML")
- PDF versions of the user guides ("PDF")
- Program Directories for IMS Tools products
- 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 <https://www.ibm.com/support/mynotifications>
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 us provide the most accurate and highest quality information. If you have any comments about this or any other IMS Tools information, see [How to provide feedback in IBM Documentation](#).

When you provide feedback, include as much information as you can about the content you are commenting on, where we can find it, and what your suggestions for improvement might be.

## 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 25](#)
- [“Configuring for initial installation” on page 26](#)
- [“Migration procedures” on page 26](#)

---

## 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 Database Utility Solution 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 Library Integrity Utilities is designed to operate with any version of z/OS that supports the version of IMS that you are running. All supported releases of IMS are supported by IMS Library Integrity Utilities.

#### Integrity Checker utility

- To activate Integrity Checker, a DBRC environment is required. Also, the DBRC RECON data sets must be generated by IMS 11.1 or later releases.
- To migrate your LICON data set from IMS Library Integrity Utilities 1.1 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 9.1 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 11 or later.
- RECON data sets that are generated by IMS 11 or later, and the IMS SDFSRESL library of IMS 11.1 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 11 or later.

**Catalog Manager utility**

Supports ACB libraries that are generated by IMS 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 36.

**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”](#) on page 26.

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 2.2, 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”](#) on page 27

- [“Migrating Advanced ACBGEN” on page 31](#)

## 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 2.1 \(non BPE-based DBRC\)” on page 27](#)
- [“Migrating from IMS Library Integrity Utilities 2.1 \(BPE-based DBRC\)” on page 28](#)
- [“Migrating from IMS Library Integrity Utilities 1.1” on page 28](#)

### Migrating from IMS Library Integrity Utilities 2.1 (non BPE-based DBRC)

When you migrate Integrity Checker from IMS Library Integrity Utilities 2.1 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 2.1 are compatible.

However, in IMS Library Integrity Utilities 2.2, 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. 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 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 2.1 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 53](#).

- If the Integrity Checker load modules were merged into the IMS SDFSRESL library in the IMS Library Integrity Utilities 2.1 environment, merge the Integrity Checker load modules of IMS Library Integrity Utilities 2.2 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 56](#).

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 2.1 (BPE-based DBRC)

When you migrate Integrity Checker from IMS Library Integrity Utilities 2.1 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 2.1 are compatible.

However, in IMS Library Integrity Utilities 2.2, 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 2.1 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 53.

- If the Integrity Checker load modules were merged into the IMS SDFSRESL library in the IMS Library Integrity Utilities 2.1 environment, merge the Integrity Checker load modules of IMS Library Integrity Utilities 2.2 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 56.

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 1.1

When you migrate Integrity Checker from IMS Library Integrity Utilities 1.1 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 1.1 are compatible. However, in IMS Library Integrity Utilities 2.2, alias name DSPCRTR0 is not assigned for the FABLRTR0 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 1.1 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 1.1 can be used in this version of the product. However, Integrity Checker of IMS Library Integrity Utilities 2.2 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 80](#)
- [“Changing the global option module ” on page 67](#)

### 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 1.1. You must complete one of the following steps:

- Migrate the LICON data sets used by IMS Library Integrity Utilities 1.1 to the new format by using the LICON data set migration utility that is provided in IMS Library Integrity Utilities 2.2.
- 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 1.1 to the format of IMS Library Integrity Utilities 2.2.

#### 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 46](#)
- [“Defining and initializing the LICON data set” on page 49](#)

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

#### c) Migrate the original LICON data set from IMS Library Integrity Utilities 1.1 to IMS Library Integrity Utilities 2.2 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 1.1. If the LICON data set is used in Integrity Checker of IMS Library Integrity Utilities 1.1, unexpected results can occur.

Use the JCL sample shown in [Figure 1 on page 30](#) 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 1.1 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 53](#).

- If the Integrity Checker load modules were merged into the IMS SDFSRESL library in the IMS Library Integrity Utilities 1.1 environment, merge the Integrity Checker load modules of IMS Library Integrity Utilities 2.2 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 56.

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.

```
//*
//JOB LIB DD DISP=SHR,DSN=HPS.SHPSLMD0
//*
//*=====
//*
//* STEP1 : ALLOCATE NEW LICON DATA SET AND BACK UP
//*
//*=====
//STEP1 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
DELETE (LIUHLQ.LICONDSN.NEW)
SET MAXCC=0
DEFINE CLUSTER (NAME(LIUHLQ.LICONDSN.BACKUP) -
INDEXED -
KEY (32 0) -
SHAREOPTIONS(3,3) -
NOREUSE -
VOL (VVVVVV) -
CYL (PP1 SS1) -
RECORDSIZE (4096 32760) -
FREESPACE (XX,YY) -
DATA (NAME(LIUHLQ.LICONDSN.DATA.BACKUP)) -
INDEX (NAME(LIUHLQ.LICONDSN.INDEX.BACKUP)))
DEFINE CLUSTER (NAME(LIUHLQ.LICONDSN.NEW) -
INDEXED -
KEY (44 0) -
SHAREOPTIONS(3,3) -
NOREUSE -
VOL (VVVVVV) -
CYL (PP2 SS2) -
RECORDSIZE (4096 32760) -
FREESPACE (XX,YY) -
DATA (NAME(LIUHLQ.LICONDSN.DATA.NEW)) -
INDEX (NAME(LIUHLQ.LICONDSN.INDEX.NEW)))
/*
//*=====
//*
//* STEP2 : BACK UP LICON DATA SET
//*
//*=====
//STEP2 EXEC PGM=IDCAMS,COND=(4,LT)
//INDD DD DISP=SHR,DSN=LIUHLQ.LICONDSN
//OUTDD DD DISP=SHR,DSN=LIUHLQ.LICONDSN.BACKUP
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
REPRO INFILE(INDD) OUTFILE(OUTDD)
/*
```

Figure 1. JCL sample to migrate a LICON data set (Part 1 of 2)



```

//*=====
//*
//* STEP3 : CONVERT V1 LICON DATA SET TO V2
//*
//*=====
//STEP3 EXEC PGM=SORT,COND=(4,LT)
//SORTOUT DD DISP=SHR,DSN=LIUHLQ.LICONDSN.NEW
//SORTIN DD DISP=SHR,DSN=LIUHLQ.LICONDSN.BACKUP
//SYSOUT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
SORT FIELDS=(5,44,BI,A)
RECORD TYPE=V
OPTION MAINSIZE=MAX
MODS E15=(FABLE15A,70000)
/*
//*=====
//*
//* STEP4 : RENAME NEW LICON DATA SET
//*
//*=====
//STEP4 EXEC PGM=IDcams,COND=(4,LT)
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
DELETE (LIUHLQ.LICONDSN)
IF MAXCC EQ 0 THEN -
DO
ALTER LIUHLQ.LICONDSN.NEW -
NEWNAME(LIUHLQ.LICONDSN)
ALTER LIUHLQ.LICONDSN.DATA.NEW -
NEWNAME(LIUHLQ.LICONDSN.DATA)
ALTER LIUHLQ.LICONDSN.INDEX.NEW -
NEWNAME(LIUHLQ.LICONDSN.INDEX)
END
/*

```

Figure 2. JCL sample to migrate a LICON data set (Part 2 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 2.2. 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 2.2.

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

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.



---

## 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 33](#)
- [“Planning for Integrity Checker configuration” on page 36](#)
- [“Activating Integrity Checker” on page 48](#)
- [“Maintaining Integrity Checker” on page 59](#)
- [“Preventing database corruption with Integrity Checker ” on page 70](#)
- [“Deactivating Integrity Checker” on page 76](#)
- [“Output from Integrity Checker” on page 78](#)
- [“Global option module generation macro” on page 79](#)
- [“LICON utility reference” on page 85](#)

---

### Integrity Checker overview

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

#### Subsections:

- [“DMB verification function” on page 33](#)
- [“How the DMB verification function works” on page 33](#)
- [“Change history report for DBDs and databases” on page 35](#)
- [“Supported database organizations” on page 35](#)
- [“Program structure” on page 36](#)

### DMB verification function

Integrity Checker supports the *DMB verification* function to prevent database corruptions that are caused by misuse of incorrect DBDLIB, ACBLIB, or IMS directory.

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), ACB libraries (ACBLIB), or IMS directory as load modules.

Every IMS job requires a DBDLIB, an ACBLIB, or IMS directory. IMS loads the DMB from the DBDLIB, the ACBLIB, or the IMS directory 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 34 shows DMB verification for IMS online applications, and Figure 4 on page 35 shows DMB verification for batch applications and utilities. The steps that follow the figures correspond to the numbers in the figures.

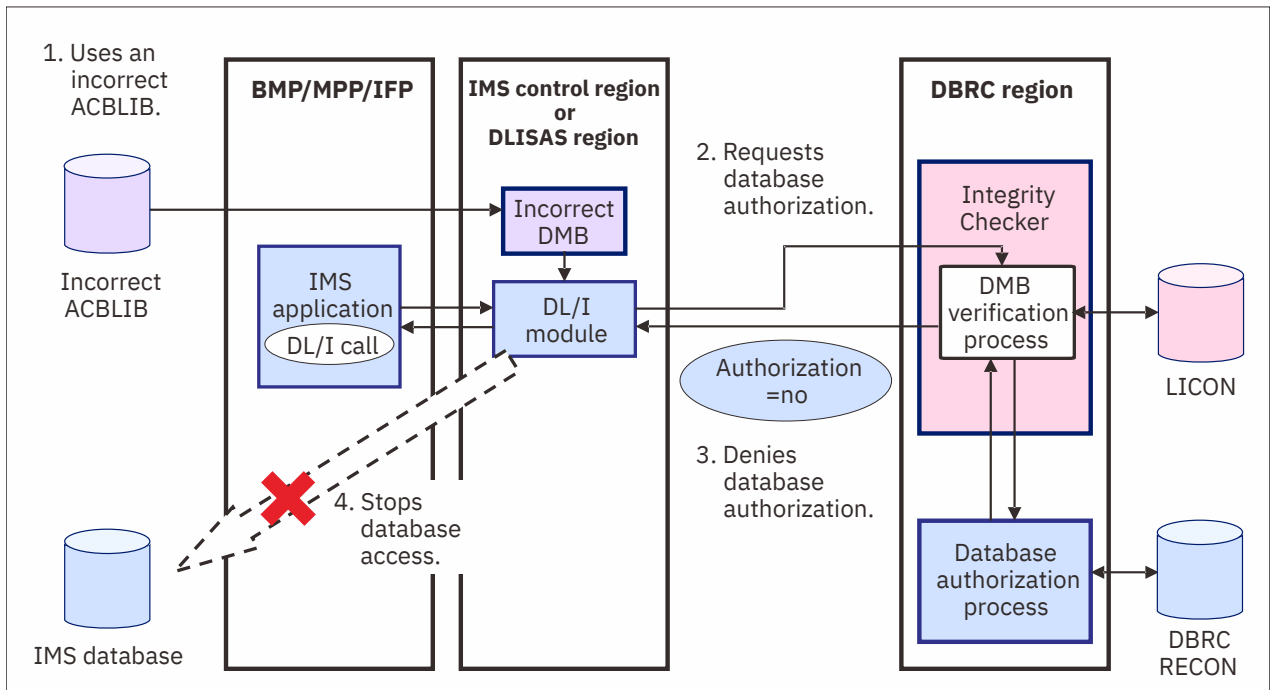


Figure 3. How Integrity Checker prevents database corruption: IMS online applications

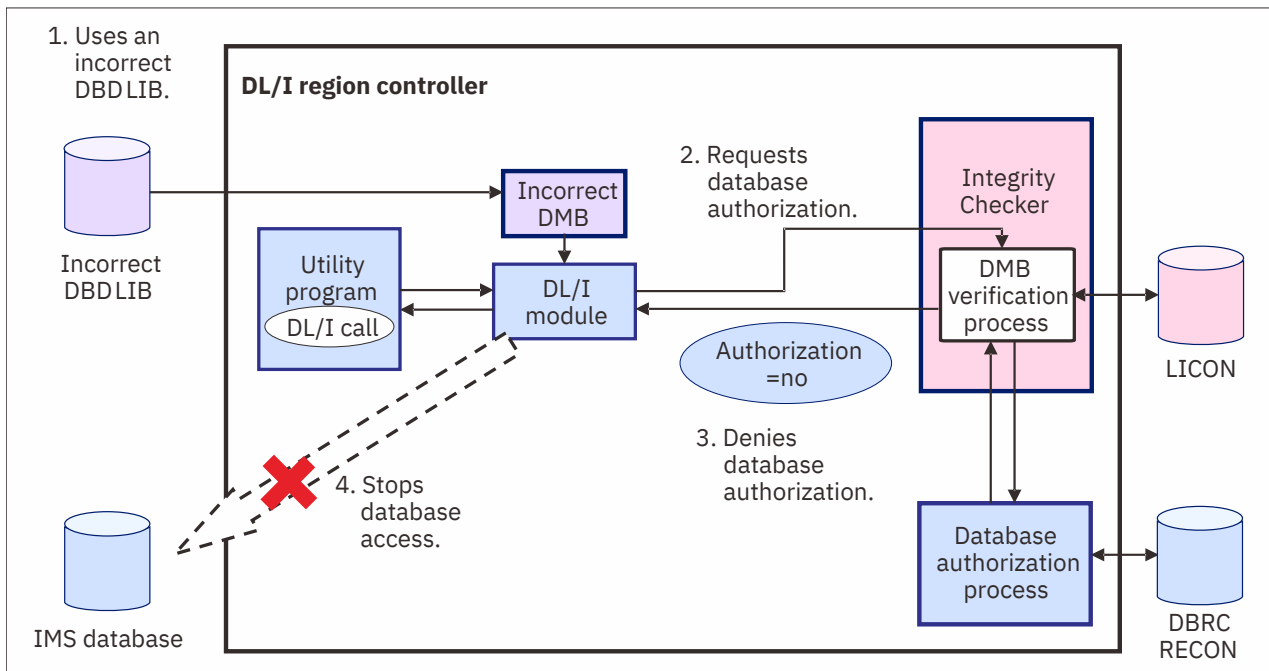


Figure 4. How Integrity Checker prevents database corruption: batch applications and utilities

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.

If the IMS management of ACBs is enabled and IMS application or utility program uses an incorrect IMS directory, Integrity Checker detects and prevents any access to the database that uses incorrect DMBs in the same manner.

## 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 48](#).

Use the following topics to design the Integrity Checker configuration:

- [“LIU load module library customization” on page 36](#)
- [“LICON data sets and global option modules” on page 37](#)
- [“Integrity Checker configuration requirements” on page 38](#)
- [“Runtime options and environments” on page 43](#)
- [“Historical data maintained in LICON data sets” on page 44](#)
- [“Considerations for activating Integrity Checker” on page 46](#)

## 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 2.1 and earlier, alias name DSPCRTR0 was assigned for the FABLRTR0 module by default, but in IMS Library Integrity Utilities 2.2 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**

*LIU 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, ACBLIB, or IMS directory 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 LIUG*imsid*.

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

- [“Single IMS subsystem configuration example” on page 39](#)
- [“Multiple IMS subsystems configuration example: Data-sharing environment” on page 39](#)
- [“Multiple IMS subsystems configuration example: Non-data-sharing environment” on page 40](#)
- [“Multiple IMS subsystems configuration example: Multiple data-sharing environments” on page 41](#)
- [“Multiple IMS subsystems configuration example: XRF complex” on page 42](#)

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 79](#).

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.

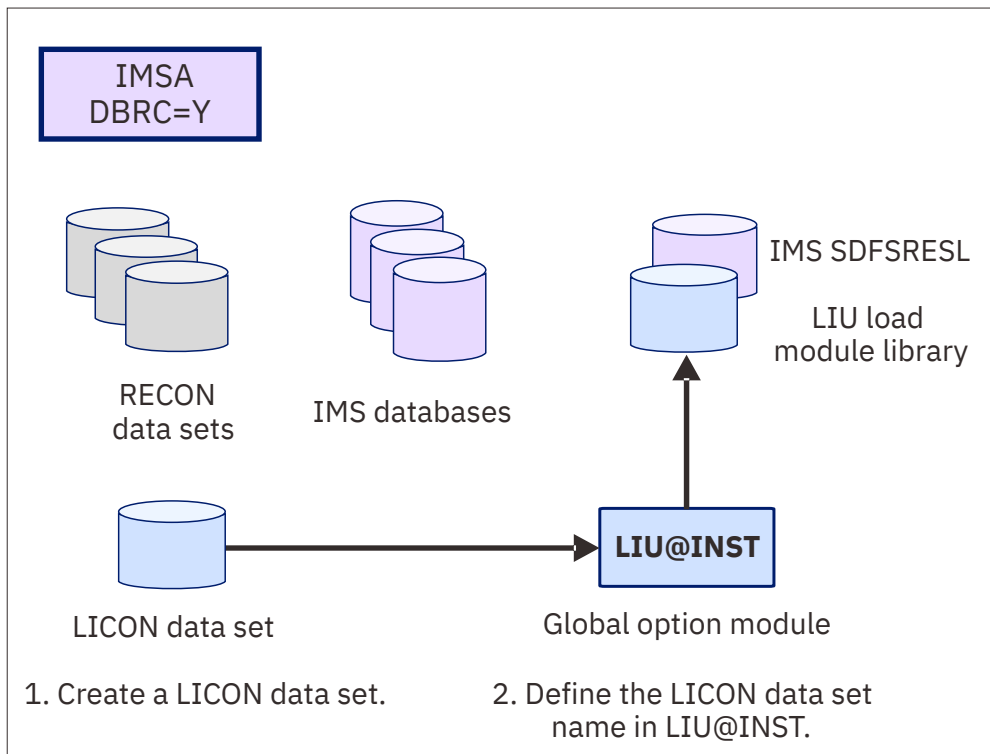


Figure 5. Integrity Checker configuration for a single IMS subsystem

### 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.

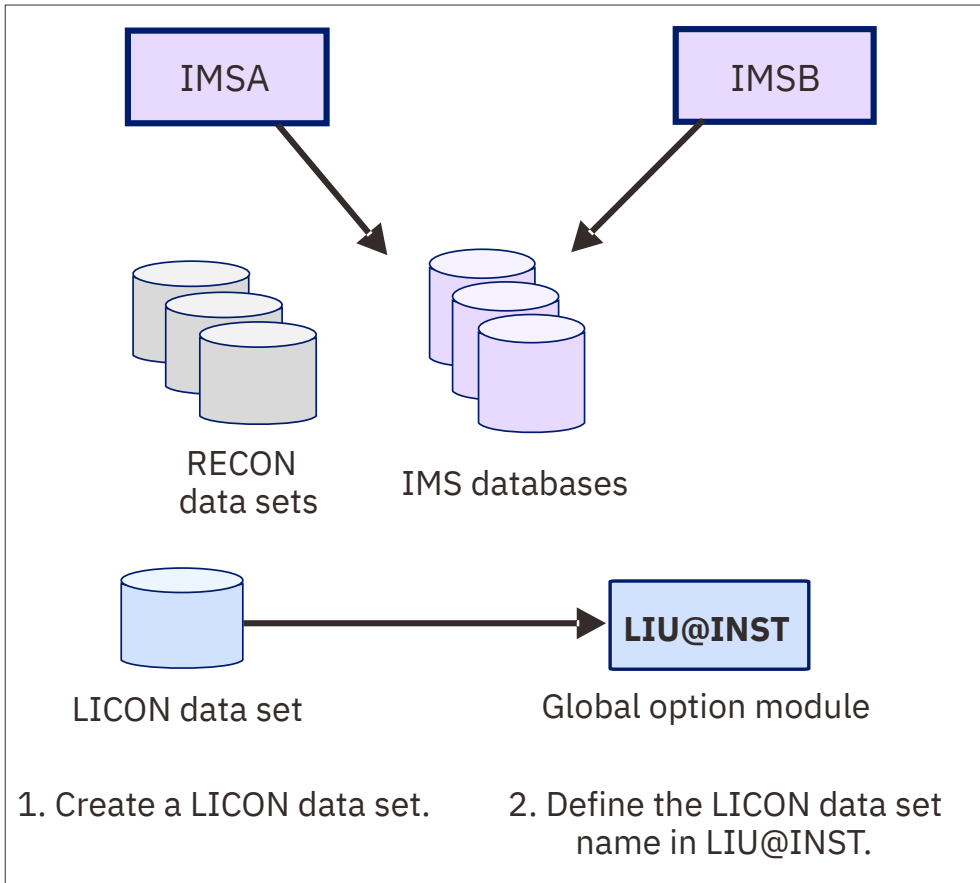


Figure 6. Integrity Checker configuration in a data-sharing environment

### 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.

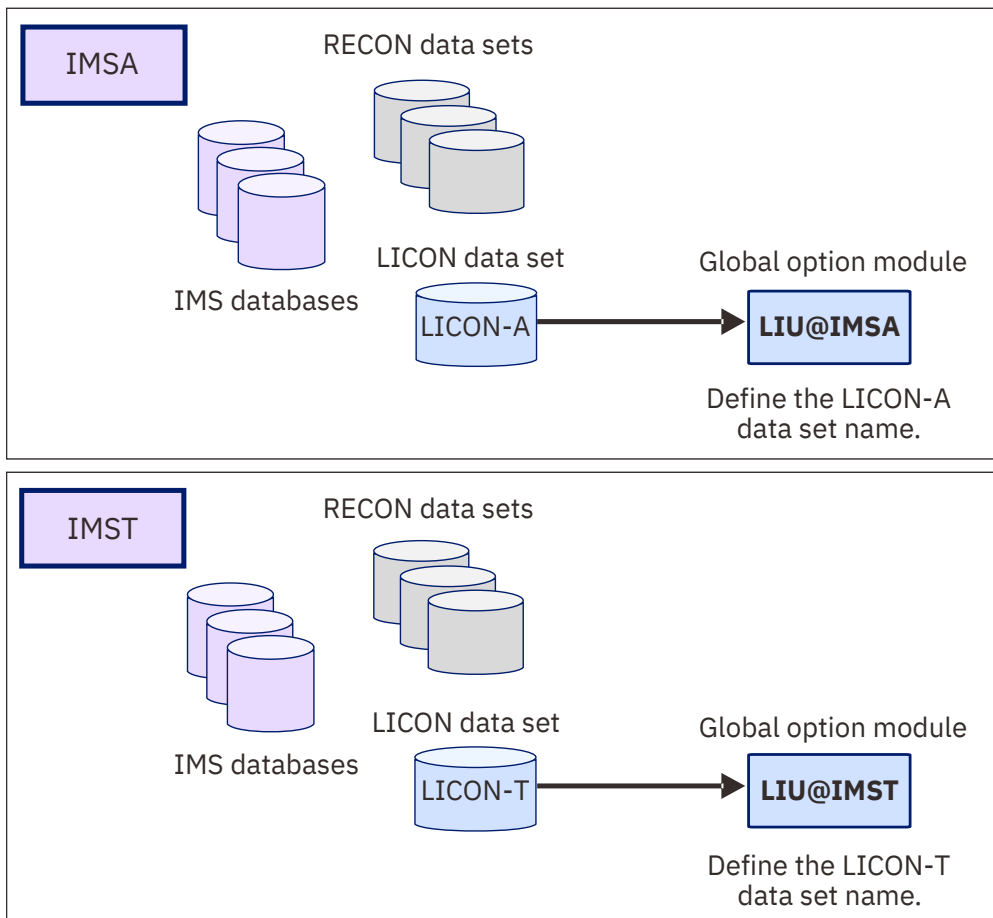


Figure 7. Integrity Checker configuration in a non-data-sharing environment

### 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.

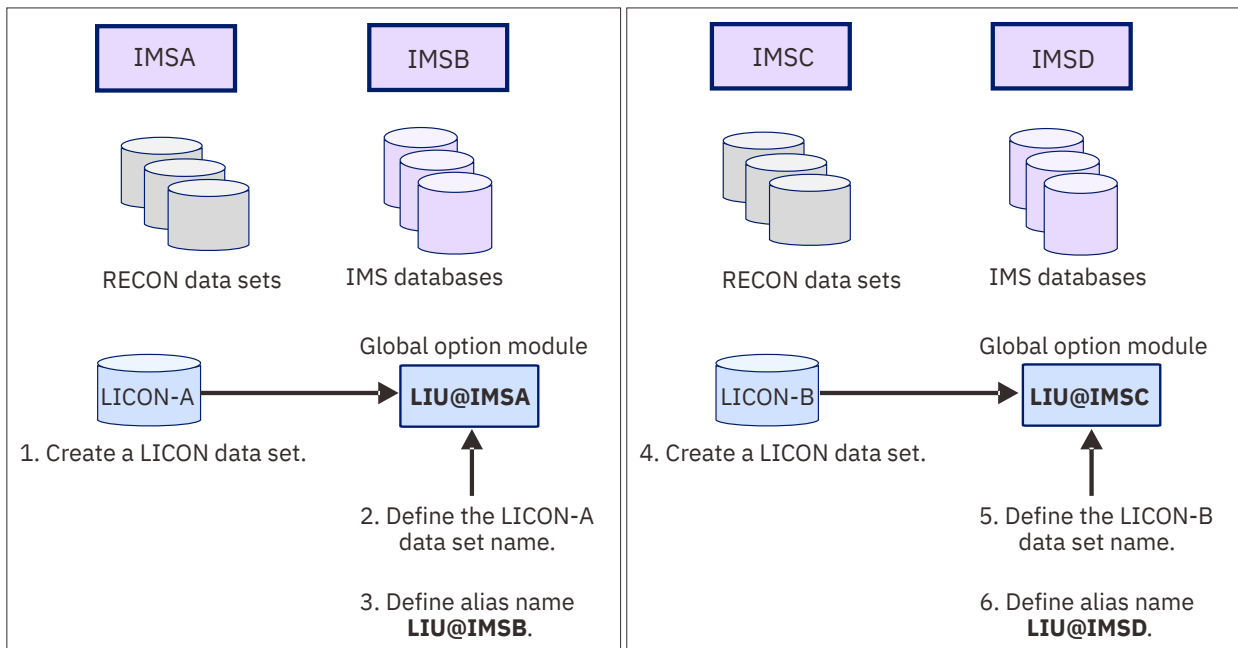


Figure 8. Integrity Checker configuration in multiple data-sharing environments

### 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.

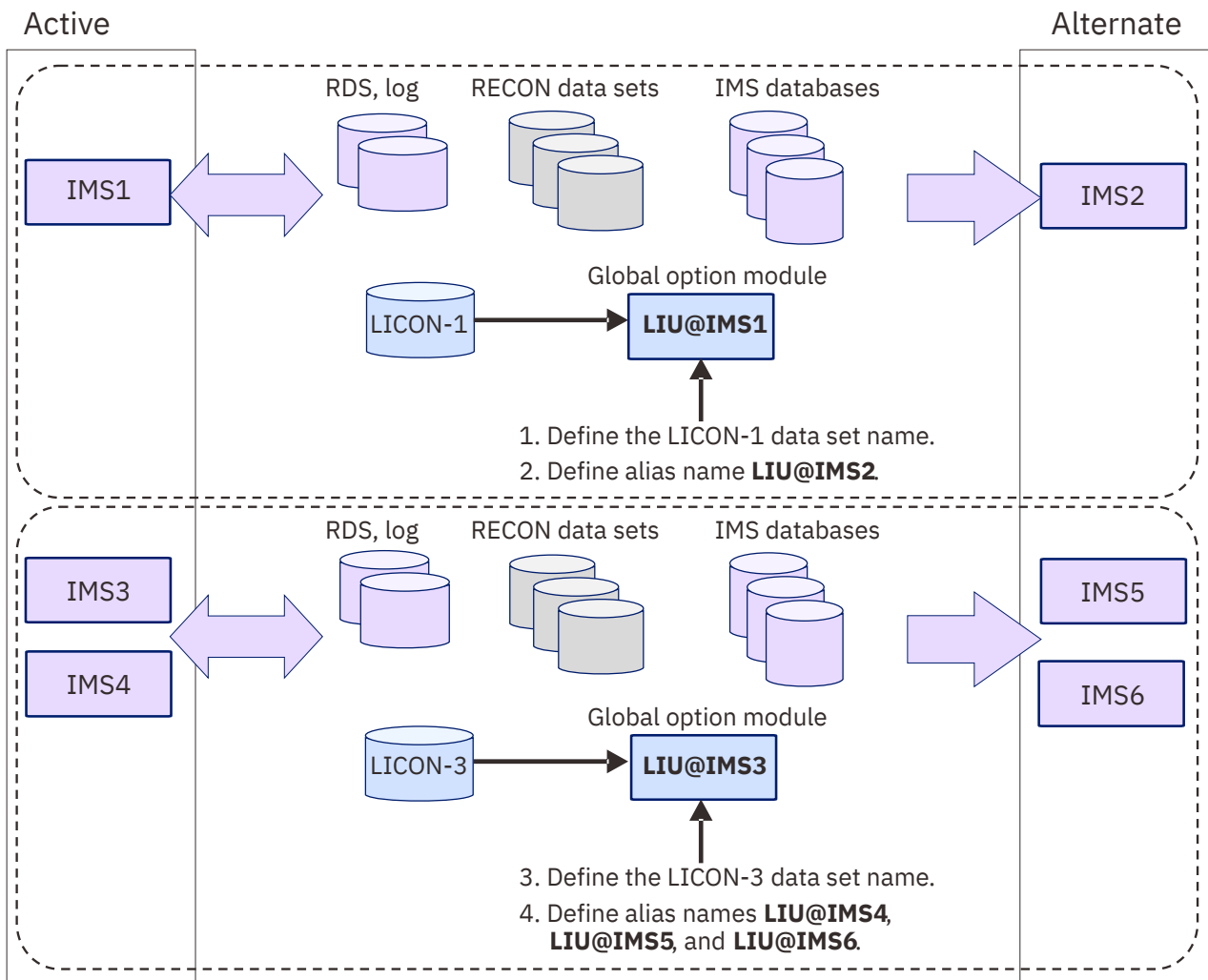


Figure 9. Integrity Checker configuration in an XRF complex

## 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

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 79](#).

## 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 102](#).
- For report field descriptions, see [“Output from Integrity Checker” on page 78](#).

## 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:

Database access	Access type identified by Integrity Checker
Load utility	Load access
Reorganization utility	Load access
Unload utility	Unload access
Recovery utility	Update access
Utility or application program with the PCB processing option (PROCOPT) of A, I, R, or D	Update access

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 79](#).
- To update the options in RDEs, see [“INIT.DB command” on page 89](#).

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 70.](#)

### 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 46.](#)

### 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 51.](#)

### 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 51.](#)

### 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 63](#)
- [“Altering the definition of a DMB verification-enabled online DEDB by using the DEDB Alter utility \(DBFUDA00\)” on page 65](#)

### 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 67.](#)

## 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:

$$\text{LICON data set size} = \text{total\_size\_required\_for\_full-function\_databases} \\ + \text{total\_size\_required\_for\_HALDBs} \\ + \text{total\_size\_required\_for\_DEDBs}$$

#### **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:

$$(450 + 32 * \text{number\_of\_DSGs} + 52 * \text{number\_of\_segments} \\ + 180 * \text{access\_info\_size}) * 16$$

**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:

$$(450 + 32 * \text{number\_of\_DSGs} + 52 * \text{number\_of\_segments} \\ + 180 * \text{access\_info\_size}) * 16$$

#### **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:

$$(220 + 48 * \text{number\_of\_segments} + 180 * \text{access\_info\_size}) * 16$$

### 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 79](#).

- 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:

$$\text{LICON data set size} = (32 * \text{number\_of\_full-function\_databases} \\ + 32 * \text{number\_of\_HALDB\_partitions} \\ + 16 * \text{number\_of\_DEDB\_areas}) \text{ KB}$$

***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, 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 36.

### About this task

To activate Integrity Checker, you first prepare the global option modules and the LICON data sets. Then, by using the DBD library, ACB library, or IMS directory, 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:

- a. [“Setting up the global option modules”](#) on page 48
- b. [“Setting up the LICON data sets”](#) on page 49
- c. [“Setting up RACF security”](#) on page 51
- d. [“Customizing LIU load modules”](#) on page 53
- e. [“Configuring for a BPE-based DBRC environment”](#) on page 57
- f. [“Verifying that Integrity Checker is activated”](#) on page 58
- g. [“Restarting IMS online and running IMS batch application, IMS utility, and IMS Tools jobs”](#) on page 58

## 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 38 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 79](#).

**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 68](#)

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=*
//SYSUT1 DD UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),
// SPACE=(1024,(100,10),RLSE),DISP=(,DELETE)
//SYSLMOD DD DISP=SHR,DSN=HPS.SHPSLMD0
//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 38](#) 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.

- a. [“Defining and initializing the LICON data set” on page 49](#)
- b. [“Creating an RDE to register DMB information” on page 50](#)
- c. [“Serializing the LICON data set” on page 51](#)

## 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 46.

```
//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.SHPSLMDO
// DD DISP=SHR,DSN=IMSVS.SDFSRESL
//FABLPRNT 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, the ACB library, or the IMS directory 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=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 *
INIT.DB DBD(*)
/*
```

## Serializing the LICON data set

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

### 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:

Parameter	Value
QNAME	'FABLICON'
RNAME	The name of the LICON data set

## 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” on page 51](#)
- [“Setting up security for IMS Database Recovery Facility jobs” on page 52](#)

## Setting up security for LICON data sets

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

### 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.

#### **ADDSD 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 36 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 DSPCRTR0"](#) on page 53
- ["Method 2. Customizing LIU load modules by merging into the IMS SDFSRESL library"](#) on page 56

### Method 1. Customizing LIU load modules by creating alias name DSPCRTR0

Create an alias name DSPCRTR0, 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 DSPCRTR0 by link-editing the FABLRTR0 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 DSPCRTR0 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 77 and you want to reactivate Integrity Checker.

If you are reactivating Integrity Checker and the DSPCRTR0 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 DSPCRTR0. This JCL is in the SHPSJCL0 library, member FABLALSC.

When the job ends, confirm that the return code is 0.

```

//FABLALSC JOB
//*-----
//*  STEP1: Add the alias DSPCRTR0
//*-----
//LINK EXEC PGM=IEWL,REGION=0M,
//  PARM='SIZE=(880K,64K),LET,LIST,NCAL,RENT,REFR,XREF'
//* IEWL = IEWBLINK
//*
//SYSPRINT DD SYSOUT=*
//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 AHPSMOD0 TO THE INCLUDE STATEMENT,
//* THE NON-ACCEPTED CHANGES WILL BE DELETED.
//SYSLIN DD *
  ENTRY FABLRT0
  INCLUDE SYSLMOD(FABLRT0)
  ALIAS DSPCRTR0
  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.

## 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.

### FABLSDAP DD

Optionally, you can specify the FABLSDAP 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.



```

//      PROC RGN=64M,DPTY='(14,15)',SOUT=A,
//      IMSID=SYS3,SYS2=,IMSPLEX=
//IEFPROC EXEC PGM=DFSMVRC0,REGION=&RGN,
//      DPRTY=&DPTY,PARM='DRC,&IMSID,IMSPLEX=&IMSPLEX'
//*****
//*
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR      <<---- STEPLIB DD
//      DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//PROCLIB DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//JCLOUT  DD SYSOUT=(A,INTRDR)
//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 10. Example of DBRC procedure for a non-BPE-based DBRC region

```

//DBRC   PROC RGN=0M,SOUT=A,
//      RESLIB='IMS.SDFSRESL',
//      BPECFG=BPECFG,
//      DBRCINIT=000,
//      IMSID=IMS1,
//      PARM1='BPEINIT=DSPBINIO'
//*
//DBRCPROC EXEC PGM=BPEINI00,REGION=&RGN,
// PARM='BPECFG=&BPECFG,DBRCINIT=&DBRCINIT,IMSID=&IMSID,&PARM1'
//*
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR      <<---- STEPLIB DD
//      DD DSN=&RESLIB,DISP=SHR
//      DD DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB DD DSN=IMS.PROCLIB,DISP=SHR
//SYSPRINT DD SYSOUT=&SOUT
//SYSUDUMP DD SYSOUT=&SOUT
//JCLOUT  DD SYSOUT=(A,INTRDR)
//JCLPDS  DD DSN=IMS.PROCLIB,DISP=SHR
//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

```

//      PROC MBR=TEMPNAME,PSB=,BUF=7,
//      SPIE=0,TEST=0,EXCPVR=0,RST=0,PRLD=,
//      SRCH=0,CKPTID=,MON=N,LOGA=0,FMTO=T,
//      IMSID=,SWAP=,DBRC=,IRLM=,IRLMNM=,
//      BKO=N,IOB=,SSM=,APARM=,
//      RGN=4M,
//      SOUT=A,LOGT=2400,SYS2=,
//      LOCKMAX=,GSGNAME=,TMINAME=,
//      IMSPLEX=
//G      EXEC PGM=DFSRR00,REGION=&RGN,
//      PARM=(DLI,&MBR,&PSB,&BUF,
//      &SPIE&TEST&EXCPVR&RST,&PRLD,
//      &SRCH,&CKPTID,&MON,&LOGA,&FMTO,
//      &IMSID,&SWAP,&DBRC,&IRLM,&IRLMNM,
//      &BKO,&IOB,&SSM,'&APARM',
//      &LOCKMAX,&GSGNAME,&TMINAME,
//      &IMSPLEX)
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR      <<---- STEPLIB DD
//      DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//      DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//DFSRESLB DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//IMS      DD DSN=IMS.&SYS2.PSBLIB,DISP=SHR
//      DD DSN=IMS.&SYS2.DBDLIB,DISP=SHR
//PROCLIB DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//IEFRDER DD DSN=IMSLOG,DISP=(,KEEP),VOL=(,,99),
//      UNIT=(&LOGT,,DEFER),
//      DCB=(RECFM=VB,BLKSIZE=4096,
//      LRECL=4092,BUFNO=2)
//IEFRDER2 DD DSN=IMSLOG2,DISP=(,KEEP),VOL=(,,99),
//      UNIT=(&LOGT,,DEFER,SEP=IEFRDER),
//      DCB=(RECFM=VB,BLKSIZE=4096,
//      LRECL=4092,BUFNO=2)
//SYSUDUMP DD SYSOUT=&SOUT,
//      DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605),
//      SPACE=(605,(500,500),RLSE,,ROUND)
//IMSMON  DD DUMMY
//FABLPRNT DD SYSOUT=&SOUT      <<---- FABLPRNT DD

```

Figure 12. Example of DLIBATCH procedure

## 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 FABLRT0 module into the IMS SDFSRESL library. The FABLUMD1 job is shown in [“JCL example to install the FABLRT0 module into the IMS SDFSRESL library”](#) on page 57.

4. Merge the LIU load modules (FABL\* members) in the target library SHPSLMD0 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

```
//FABLUMD1 JOB
//*-----
/* STEP1: SMP/E RECEIVE/APPLY usermod ZZLIU01 to IMS CSI
//*-----
//STEP1 EXEC procedure name of IMS SMP/E job
/*
/*
//AHPSMOD0 DD DISP=SHR,DSN=LIU.AHPSMOD0
/*
//SMPPTFIN DD DATA,DLM=@@
++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 CALLLIBS /* JCLIN FOR LIU MODULES */
/*
//LINK EXEC PGM=IEWL,REGION=0M,
// PARM='SIZE=(880K,64K),LET,LIST,NCAL,RENT,REFR,XREF'
//SYSPRINT DD SYSOUT=A
//AHPSMOD0 DD DISP=OLD,DSN=LIU.AHPSMOD0
//ADFSLOAD DD DISP=OLD,DSN=IMS.ADFSLOAD
//SYSLMOD DD DISP=OLD,DSN=IMS.SDFSRESL
//SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(10,1))
//SYSLIN DD *
    INCLUDE AHPSMOD0(FABLRTR0)
    INCLUDE ADFSLOAD(DSPCRTR0)
    ENTRY FABLRTR0
    NAME DSPCRTR0(R)
/*
++MOD (FABLRTR0) LKLIB(AHPSMOD0) /* LIU MODULE MOD ENTRY */
@@
//SMPCNTL DD *
    SET BDY (GLOBAL).
    RECEIVE S (ZZLIU01) SYSMODS.
    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](#)

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

## Maintaining Integrity Checker

---

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

The following topics explain the Integrity Checker maintenance tasks:

- [“Maintaining RDEs” on page 59](#)
- [“Maintaining global option modules” on page 67](#)
- [“Maintaining LICON data sets” on page 69](#)
- [“Restarting Integrity Checker after an abend” on page 69](#)
- [“Applying PTFs to IMS Library Integrity Utilities and to IMS” on page 69](#)

**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” on page 59](#)
- [“RDE maintenance at database reorganization” on page 60](#)
- [“RDE maintenance at database recovery” on page 62](#)
- [“RDE maintenance at DBD or RECON change” on page 63](#)

### 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 the IMS management of ACBs is enabled, use the INIT.DB command of the LICON utility to manually create an RDE by specifying the IMS directory.

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. If the IMS management of ACBs is enabled and when IMS Online Reorganization Facility is used to reorganize the database, the ONLINECHANGE(N) option is forced. After the IMPORT DEFN SOURCE(CATALOG) command activates the new database definition, use the CHANGE.DB command of the LICON utility to manually re-create the RDE by specifying the IMS directory.

## 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.

### **Considerations when reorganizing databases in the IMS management of ACBs environment**

After you reorganize databases to change the DBD definition by using the IMS directory staging data set, you must activate the new database definition by issuing the IMPORT DEFN SOURCE(CATALOG) command. Before you start IMS online, you need to re-create the RDE with the IMS directory 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.

If the IMS management of ACBs is enabled, re-create the RDE by specifying the IMS directory after the IMPORT DEFN SOURCE(CATALOG) command is issued.

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.

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

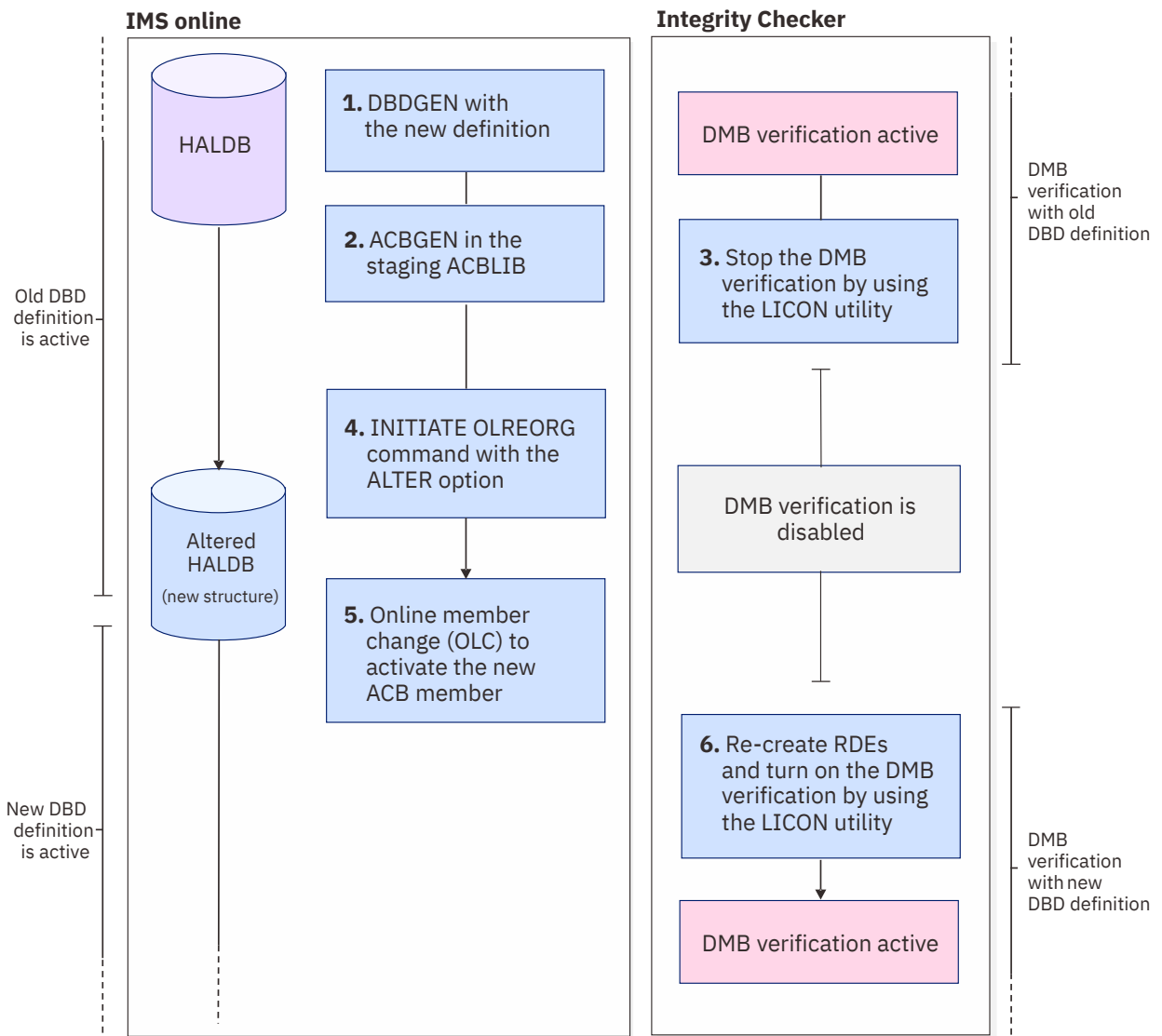


Figure 13. Steps to alter an online HALDB that has DMB verification turned on

## 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:

```
CHANGE .DB DBD(dbname) CHECKON(N)
```

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 *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 *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:

```
INIT.DB DBD(dbname) CHECKON(Y,D) REPLACE
```

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.

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

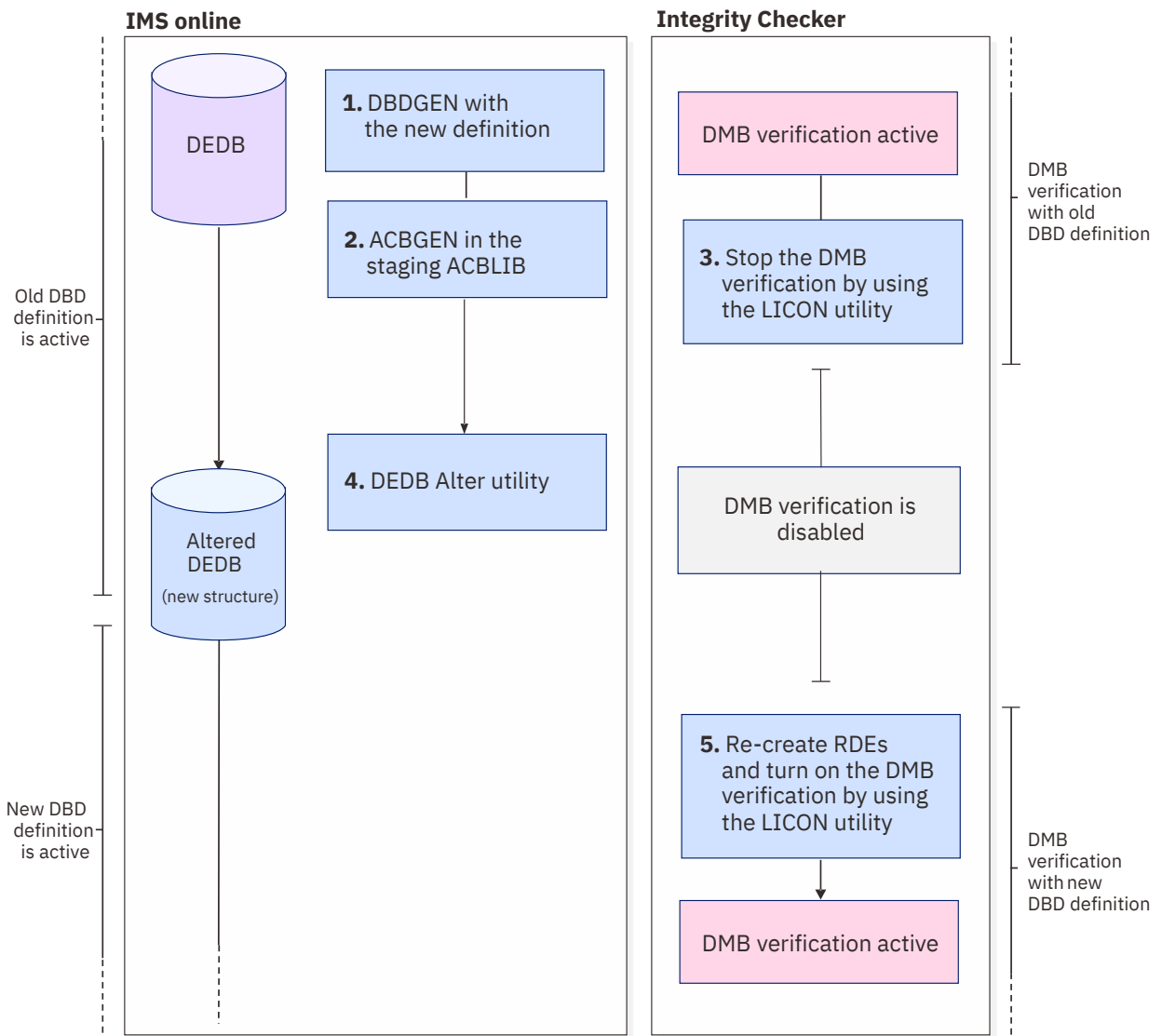


Figure 14. Steps to alter an online DEDB that has DMB verification turned on

## 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:

Changed definition	LICON utility command
Altering a DEDB area	<code>CHANGE.DB DBD(dbname) AREA(areaname) CHECKON(N)</code>
Replacing the randomizer	<code>CHANGE.DB DBD(dbname) CHECKON(N)</code>

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*.

- 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	LICON utility command
Altering a DEDB area	<code>INIT.DB DBD(<i>dbname</i>) AREA(<i>areaname</i>) CHECKON(Y,D) REPLACE</code>
Replacing the randomizer	<code>INIT.DB DBD(<i>dbname</i>) CHECKON(Y,D) REPLACE</code>

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 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 ” on page 67](#)
- [“Options applied to RDEs when multiple global option modules exist with different effective ranges” on page 68](#)

### **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](#)

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.

### Related reference

[DELETE.DB command](#)

The DELETE.DB command causes the specified RDEs to be deleted.

## 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 LIUG*imsid* 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 LIUG*imsid* 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 LIUG*imsid*, Integrity Checker ignores LIUG*imsid* 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 FABLRTR0 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” on page 70](#)
- [“DMB mismatch in IMS online environment or application jobs” on page 71](#)
- [“DMB mismatch during database maintenance and operation” on page 74](#)
- [“Addressing a DMB mismatch” on page 75](#)

### 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 Preorganization utility (DFSURPRO) (for non-HALDBs)
- Database Change Accumulation utility (DFSUCUM0)
- Batch Backout utility (DFSDB000)
- HISAM Reorganization Unload 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” on page 71](#)
- [“DMB verification methods” on page 72](#)
- [“Actions when a DMB mismatch occurs” on page 72](#)

### 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, the ACBLIB, or the IMS directory 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.

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

## 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 ends abnormally 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:

```
FABL0208W CHECKSUM MISMATCH FOUND FOR DBD: dbdname
FABL0208W Type of the user exit routine
FABL0208W MODULE NAME: name of the use exit routine
```

After issuing the warning messages, Integrity Checker re-creates the RDE and issues the following informational message:

```
FABL0201I RDE CREATED FOR DBD: dbdname
```

In creating a new RDE, Integrity Checker uses the DMB information that is being used at the time when the RDE is created. The original RDE, which was valid until the time the new RDE was created, expires and is kept as the most recent historical copy of the RDE. If the maximum number for keeping expired RDEs exceeds, Integrity Checker deletes the oldest 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 59](#).

Subsections:

- [“Initial database load” on page 74](#)
- [“Database reorganization” on page 74](#)
- [“Database recovery” on page 74](#)
- [“DBD change” on page 75](#)

### 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 59](#).

### 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, IMS directory, or a load library that contains incorrect user exit routine is used. To resolve the problem, see [“Addressing a DMB mismatch” on page 75](#).

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, IMS directory, or a load library that contains incorrect user exit 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” on page 75](#).

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 60](#).

### 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, IMS directory, 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, IMS directory, 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” on page 75](#).

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 62.

## DBD change

If a DMB mismatch is found after a DBD change, ensure that an incorrect DBDLIB, IMS directory, 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, IMS directory, 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 IMS directory, 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 IMS directory, 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”](#) on page 75.

## Addressing a DMB mismatch

When Integrity Checker notifies you about a mismatch, it means that an incorrect DBDLIB, ACBLIB, IMS directory, 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 IMS directory, 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, IMS directory, 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, the IMS directory, 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 the IMS management of ACBs is enabled, you can compare active ACBs in the IMS directory or staging ACBs in the staging data set against the RDE and identify the ACB that matches the RDE.

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, IMS directory, or RECON data sets**

Identify the mismatching element in the DBDLIB, the ACBLIB, the IMS directory, 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, the IMS directory, 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 IMS Administration Foundation features to view the content of the DBDLIB.
- Use the Catalog Manager function to print the content of the IMS directory data sets or the staging data set.
- 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, the IMS directory, or the RECON data sets. Then, run DBDGEN, PSBGEN, ACBGEN, or issue a DBRC command and rerun the IMS job. If the IMS management of ACBs is enabled, confirm whether active ACBs in the IMS directory are valid. If staging ACBs in the staging data set are valid, issue IMPORT DEFN SOURCE(CATALOG) and activate ACBs in the staging data set.

### **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 DSPCRTR0 is defined for the LIU FABLRTR0 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 77](#)
- [“Deactivating Integrity Checker when IMS Library Integrity Utilities is installed as a component of an IMS tools solution pack” on page 77](#)

## 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 77 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”](#) on page 78 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 an IMS 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 SHPSJCLO library, member FABLALSD.

```
//FABLALSD JOB
//*-----*/
//* STEP1: Scratch the alias DSPCRTR0 from SHPSLMD0
//*-----
//STEP1 EXEC PGM=IEHPRGM
//*
//*
//SYSPRINT DD SYSOUT=*
//liuvol DD DISP=SHR,UNIT=SYSALLDA,VOL=SER=liuvol
//SYSIN DD *
SCRATCH MEMBER=DSPCRTR0,VOL=SYSALLDA=liuvol, C
          DSN=LIU.SHPSLMD0 LIU target load module lib
/*
```

Figure 15. Removing the alias name DSPCRTR0

## 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.

```
18086 20:09:58.63 FABL0101I LIU INTEGRITY CHECKER NOW ACTIVE WITH LICON: HLQ.IMS1.LICON
18086 20:09:58.63 FABL0102I LIU INTEGRITY CHECKER INITIALIZATION COMPLETED
18086 20:09:58.63 FABL0114I LIU INTEGRITY CHECKER ACTIVATED. IMS VERSION IS 15
```

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.



```

18120 14:32:15.42 FABL0101I LIU INTEGRITY CHECKER NOW ACTIVE WITH LICON: HLQ.IMS1.LICON
18120 14:32:15.42 FABL0102I LIU INTEGRITY CHECKER INITIALIZATION COMPLETED
18120 14:32:15.42 FABL0114I LIU INTEGRITY CHECKER ACTIVATED. IMS VERSION IS 15
18120 14:32:18.61 FABL0203W DMB MISMATCH FOUND FOR DBD: DBTEST1
18120 14:32:18.61 FABL0203W NUMBER OF SEGMENT TYPES
18120 14:32:18.61 FABL0203W RDE VALUE: 3
18120 14:32:18.61 FABL0203W ACB VALUE: 2
18120 14:32:18.61 FABL0205E VERIFICATION PROCESS FOR DBTEST1 HAS BEEN STOPPED
18120 14:32:18.61 FABL0205E REASON: SEVERE DMB MISMATCH FOUND
18120 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.

```

18120 15:26:14.88 FABL0101I LIU INTEGRITY CHECKER NOW ACTIVE WITH LICON: HLQ.IMS1.LICON
18120 15:26:14.88 FABL0102I LIU INTEGRITY CHECKER INITIALIZATION COMPLETED
18120 15:26:14.88 FABL0114I LIU INTEGRITY CHECKER ACTIVATED. IMS VERSION IS 15
18120 15:26:17.69 FABL0204E DMB MISMATCH FOUND FOR DBD: DBTEST1
18120 15:26:17.69 FABL0204E NUMBER OF SEGMENT TYPES
18120 15:26:17.69 FABL0204E RDE VALUE: 3
18120 15:26:17.69 FABL0204E ACB VALUE: 2
18120 15:26:17.69 FABL0205E VERIFICATION PROCESS FOR DBTEST1 HAS BEEN STOPPED
18120 15:26:17.69 FABL0205E REASON: SEVERE DMB MISMATCH FOUND

```

Figure 18. Messages when the verification option is (Y,D) and a mismatch is found

## FABLSNAP data set

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 48](#).

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=`
- `MSGROUT=`
- `MSGDESC=`
- `VERIFYLMT=`
- `RDEBUILD=`
- `INITERR=`

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](#)

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.

## 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” on page 81](#)
- [“DD statements” on page 81](#)
- [“Control statement keywords” on page 81](#)
- [“Example” on page 85](#)

## 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 LIUG*imsid*.

### 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.

### SYSMOD 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 LIUG*imsid* (in the example in [Figure 19 on page 85](#), 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)
- FABLWMO
- FABLAIO
- 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:

**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=**

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.

**RECU=**

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 LIUG*imsid*. 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 FABLPGEN 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=imshlq.licondsn
                END
/*
//L.SYSLMOD DD DISP=SHR,DSN=HPS.USERLIB(&MBR)
```

Figure 19. JCL for creating a global option module LIU@INST

## LICON utility reference

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, ACB library, or IMS directory 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” on page 86](#)
- [“Summary of DD names” on page 86](#)
- [“DD statements” on page 87](#)
- [“JCL example” on page 87](#)

### 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 (DFSV000) 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*

DDNAME	Use	Format	Need
JOBLIB or STEPLIB	Input	PDS	Required
DFSRESLB	Input	PDS	Required
FABLICON	Input and output	KSDS	Required (see Note <a href="#">1</a> )
FABLPRNT	Output	SYSOUT	Required
FABLIN	Input	SYSIN	Required
DBDLIB	Input	PDS	Optional (see Note <a href="#">2</a> )
ACBLIB	Input	PDS	Optional (see Note <a href="#">2</a> )
RECONx	Input		Optional (see Note <a href="#">1</a> )



---

Table 3. DD names for the LICON utility (continued)

---

DDNAME	Use	Format	Need
--------	-----	--------	------

**Notes:**

1. If dynamic allocation is used, omit the DD statement.
2. If IMS directory is not specified, 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 CHKFPSEL=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)
- The SGLXLOAD library of IMS Tools Base 1.7 or later if you use IMS directory (optional)

### DFSRESLB DD

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

### FABLICON 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 85.

## Input for the LICON utility

The input for the LICON utility must be provided as FABLIN parameters.

Subsections:

- [“Runtime options” on page 88](#)
- [“Commands” on page 88](#)
- [“Control statement syntax” on page 88](#)

### Runtime options

The following runtime options are supported:

- IMSCATHLQ=*catalog\_hlq*
- IMSCAT=DIR\_ACT|DIR\_STG

### 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.

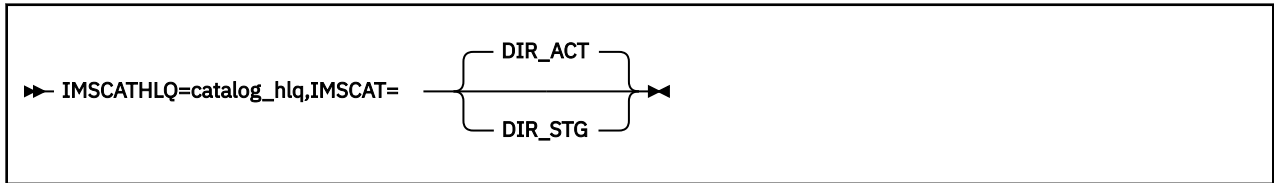
## Runtime options

When you want to use IMS directory as input for the LICON utility, specify the runtime options.

Subsections:

- [“Syntax” on page 89](#)
- [“Parameters” on page 89](#)

## Syntax



**Note:** You must specify this statement at the top of the control statement when you use IMS directory as input for the LICON utility.

## Parameters

### **IMSCATHLQ=**

Specifies the high-level qualifier of the IMS directory.

### **IMSCAT=**

Specifies either of the following values to indicate whether to use active ACBs or staging ACBs in the IMS directory:

#### **DIR\_ACT**

Specifies to use active ACBs in the IMS directory. This is the default value.

#### **DIR\_STG**

Specifies to use staging ACBs in the IMS directory staging data set.

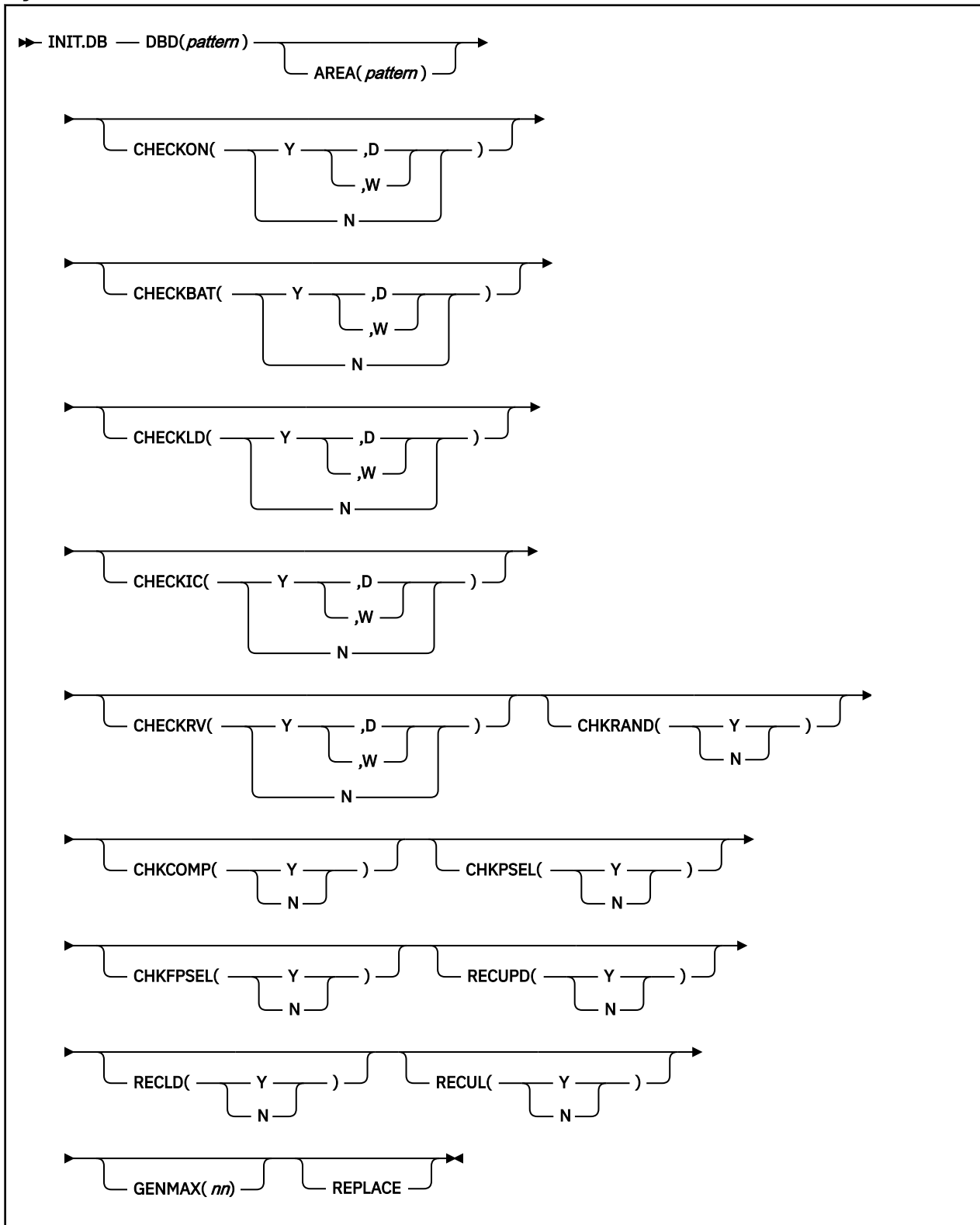
## INIT.DB command

INIT.DB command creates an RDE for one, some, or all of the DBDs or ACBs in the specified DBD or ACB library. If the LICON utility finds IMSCATHLQ and IMSCAT parameters that are specified in the FABLIN DD statement, INIT.DB command creates an RDE by using the ACBs in the specified IMS directory.

Subsections:

- [“Syntax” on page 90](#)
- [“Parameters” on page 90](#)
- [“IMS environments to be covered” on page 93](#)
- [“Response to a mismatch” on page 94](#)
- [“Examples” on page 94](#)

## 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 FABLO451E.

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:

**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.

**RECU (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 *
// INIT.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 *
// INIT.DB DBD(*)
/*
```

### INIT.DB with active ACBs in the IMS directory

This example creates an RDE for every DBD member in the IMS directory whose high-level qualifier is IMSVS.DFSCD000.

```
//LICJOB JOB
// EXEC PGM=FABLIU00
//STEPLIB DD DISP=SHR,DSN=HPS.SHPSLMD0
// DD DISP=SHR,DSN=IMSVS.SDFSRESL
// DD DISP=SHR,DSN=ITB.SGLXLOAD
//FABLPRNT DD SYSOUT=*
//FABLIN DD *
// IMSCATHLQ=IMSVS.DFSCD000,IMSCAT=DIR_ACT
// INIT.DB DBD(*)
/*
```



## INIT.LICON command

INIT.LICON command initializes the LICON data set so that it can be used.

Subsections:

- [“Syntax” on page 95](#)
- [“Parameters” on page 95](#)
- [“Example” on page 95](#)

### Syntax

```
▶▶ 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 (DFSV000) 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.

```
//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  
/*
```

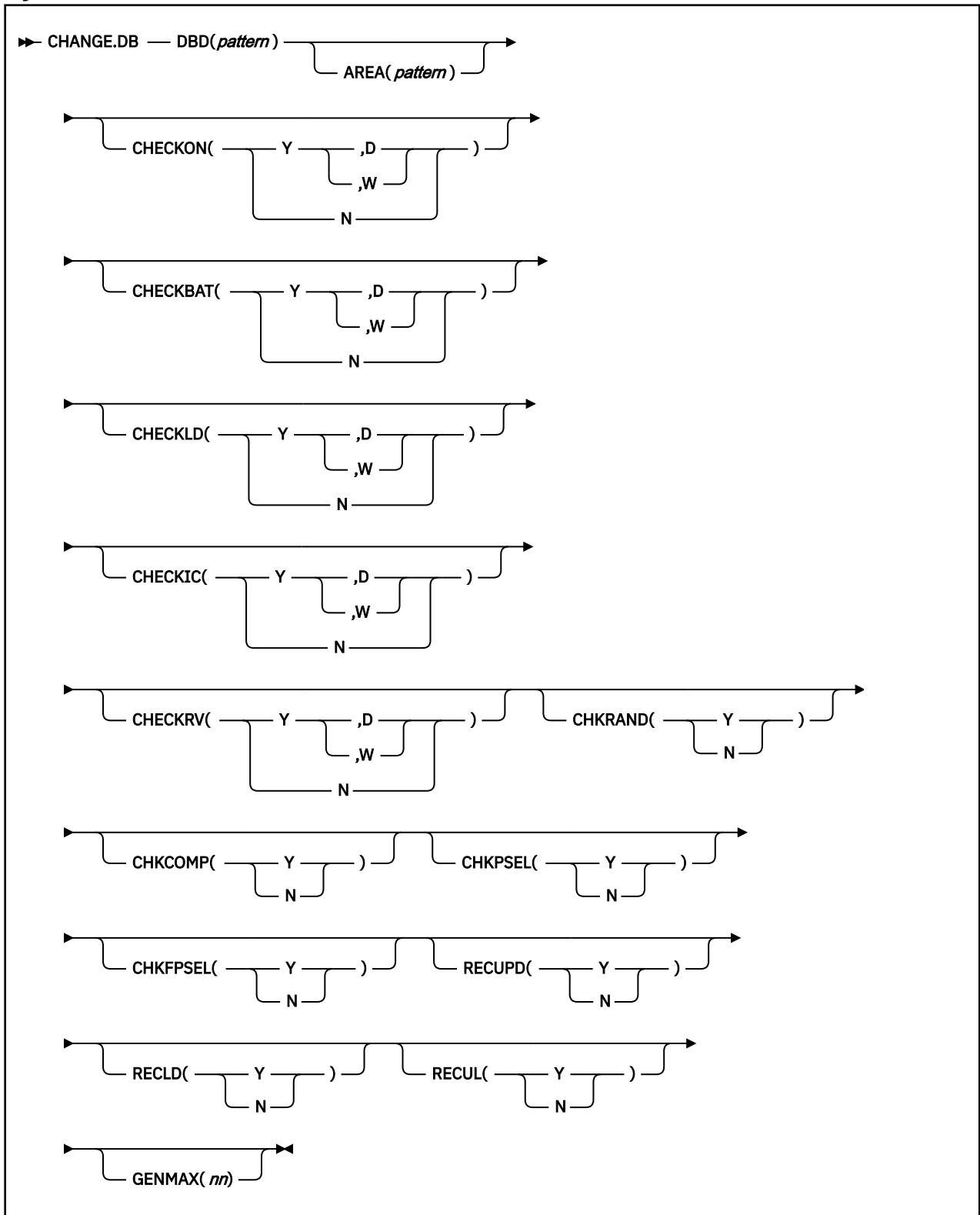
## CHANGE.DB command

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 96](#)
- [“Parameters” on page 96](#)
- [“Example” on page 99](#)

## Syntax



## 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.

**RECU (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 RECU 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.

```

//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)
/*

```

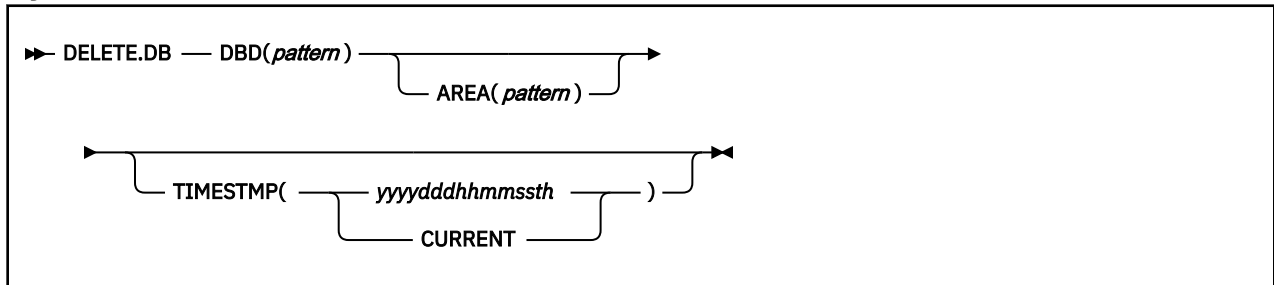
## DELETE.DB command

The DELETE.DB command causes the specified RDEs to be deleted.

Subsections:

- [“Syntax” on page 100](#)
- [“Parameters” on page 100](#)
- [“Example” on page 101](#)

### Syntax



### 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(*yyyyddhhmmssst*)

Specifies the local time stamp value of the RDE you want to delete. You can use the fully specified local time stamp *yyyyddhhmmssst* 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.

```
//LICJOB   JOB
//        EXEC PGM=FABLIU00
//STEPLIB DD DISP=SHR,DSN=HPS.SHPSLM00
//        DD DISP=SHR,DSN=IMSVS.SDFSRESL
//DBDLIB  DD DISP=SHR,DSN=IMSVS.DBDLIB
//FABLPRNT DD SYSOUT=*
//FABLIN   DD *
//        DELETE.DB DBD(DH41*)
/*
```

## 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” on page 101](#)
- [“Parameters” on page 101](#)
- [“Example” on page 102](#)

### 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
//FABLPRNT DD SYSOUT=*
//FABLIN   DD *
//          EXPIRE.DB DBD(DH41*)
/*
```

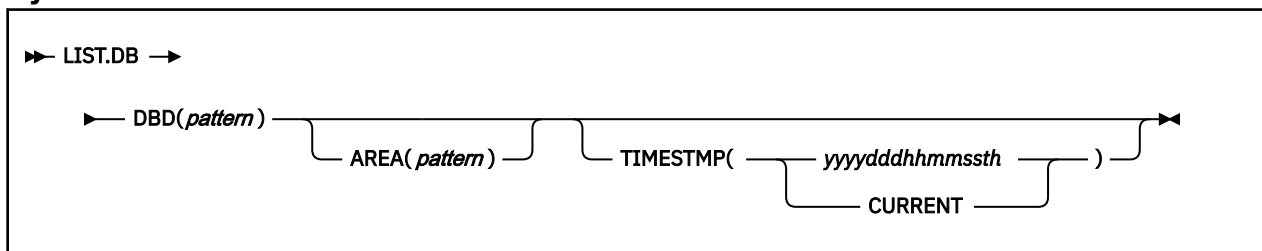
## LIST.DB command

The LIST.DB command shows you the contents of the specified RDEs, with a report.

Subsections:

- [“Syntax” on page 102](#)
- [“Parameters” on page 102](#)
- [“Example” on page 103](#)

### Syntax



### 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.

### **TIMESTMP(yyyydddhhmmssth)**

Specifies the local time stamp value of the RDE you want to list. You can use a fully specified local time stamp *yyyydddhhmmssth* 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
//         DD DISP=SHR,DSN=IMSVS.SDFSRESL
//ACBLIB   DD DISP=SHR,DSN=IMSVS.ACBLIB
//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 103](#)
- [“Parameters” on page 103](#)
- [“Example” on page 103](#)

### **Syntax**

```
►► LIST.LICON ◄◄
```

### **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 (DFSV000) 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.LICON with ACBLIB

This example lists the current RDEs for all the databases in the LICON data set.

```
//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 *
//         LIST.LICON
//*
```

## RECOVER.DB command

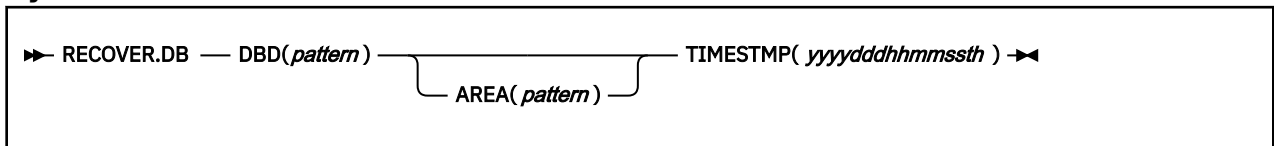
The RECOVER.DB command causes the specified RDE to be recovered by use of the expired RDE specified.

When you issue the RECOVER.DB command, the current RDE expires and the RDE specified by the TIMESTMP parameter becomes current.

Subsections:

- [“Syntax” on page 104](#)
- [“Parameters” on page 104](#)
- [“Example” on page 105](#)

### Syntax



### 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(*yyydddhhmssst*)

Specifies the local time stamp value of the RDE with which you want to recover the current RDE. Specify the entire local time stamp, *yyydddhhmssst*.

## 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 202106511301302 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.

```
//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=*
//FABIN    DD *
//          RECOVER.DB DBD(DH41TS01) TIMESTMP(202106511301302)
/*
```

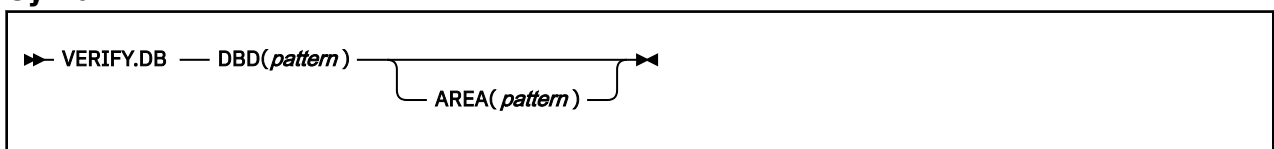
## VERIFY.DB command

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. If the LICON utility finds IMSCATHLQ and IMSCAT parameters that are specified in the FABLIN DD statement, VERIFY.DB command verifies the ACBs in the IMS directory against the RDEs stored in the LICON data set.

Subsections:

- [“Syntax” on page 105](#)
- [“Parameters” on page 105](#)
- [“Examples” on page 106](#)

## Syntax



## 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.

When you specify IMSCATHLQ and IMSCAT parameters in the FABLIN DD statement and both ACBLIB and DBDLIB DD statements are specified in your JCL, the ACBs in the DBDLIB, ACBLIB, and IMS directory are verified in the RDEs stored in the LICON data set.

## Examples

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.

```
//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
//ACBLIB DD DISP=SHR,DSN=IMSVS.ACBLIB
//FABLPRNT DD SYSOUT=*
//FABLIN DD *
//      VERIFY.DB DBD(*)
/*
```

### VERIFY.DB with DBDLIB, ACBLIB and IMS directory

This example verifies all current RDEs in the LICON data set.

```
//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
//ACBLIB DD DISP=SHR,DSN=IMSVS.ACBLIB
//FABLPRNT DD SYSOUT=*
//FABLIN DD *
//      IMSCATHLQ=IMSVS.DFSCD000,IMSCAT=DIR_ACT
//      VERIFY.DB DBD(*)
/*
```

## 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.

```

IMS LIBRARY INTEGRITY UTILITIES - LICON UTILITY                PAGE 0001
  LIST.DB DBD( *          )
2021.274 17:05:00.87          "LISTING OF RDE"                PAGE 0002
-----
RDE KEY:
  DBD=DBDLIUIV TIMESTAMP=** CURRENT **
...
...
FABL0441I DATABASE: DBDLIUIV TimestMP: 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” on page 107](#)
- [“Report field descriptions” on page 109](#)

## Sample report

The following figure shows an example of the listing of RDE.

LIST.DB DBD(\*) )

2021.274 20:16:13.23

"LISTING OF RDE"

PAGE 0002

-----  
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 = 2021.274 16:16:12.479195 (LOCAL)  
 = 2021.274 07:16:12.479195 (UTC)  
 LATEST CHANGE DATE OF RDE = N/A (LOCAL)  
 = N/A (UTC)  
 RDE FORMAT LEVEL = 2.2  
 RECOVERABLE WITH TIMESTAMP= Y

## RECORDED ACCESS INFORMATION SECTION:

LATEST ACCESS FOR UPDATE  
 DATE = 2021.274 17:37:48.404348 (LOCAL)  
 = 2021.274 08:37:48.404348 (UTC)  
 SUBSYSTEM NAME = JOBUPD  
 LATEST ACCESS FOR LOAD  
 DATE = 2021.274 17:30:52.619082 (LOCAL)  
 = 2021.274 08:30:52.619082 (UTC)  
 SUBSYSTEM NAME = JOBLD  
 LATEST ACCESS FOR UNLOAD  
 DATE = 2021.274 17:31:05.178342 (LOCAL)  
 = 2021.274 08:31:05.178342 (UTC)  
 SUBSYSTEM NAME = JOBUL

*Figure 21. Listing of RDE (HDAM) (Part 1 of 2)*

DMB SECTION:

```
DB ORGANIZATION           = HDAM/VSAM
NUMBER OF DATA SET GROUPS =      2
NUMBER OF SEGMENTS        =      3
DATXEXIT USED              = NO
VERSION ID LENGTH         =     13
VERSION ID                 = 10/01/2119.10
```

DACS:

```
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
```

...

Figure 22. Listing of RDE (HDAM) (Part 2 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.

#### 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 113](#)
- [“Restriction for Consistency Checker” on page 115](#)
- [“Checking the consistency of definitions ” on page 115](#)
- [“JCL requirements for the Consistency Checker utility” on page 116](#)
- [“Control statements for the Consistency Checker utility” on page 118](#)
- [“JCL examples for the Consistency Checker utility” on page 120](#)
- [“Output from the Consistency Checker utility” on page 121](#)

### 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” on page 113](#)
- [“Program structure” on page 114](#)
- [“Data flow” on page 114](#)

### 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 (RDDs)
- 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 organization defined in the specified DBD, as shown in the following table.

Table 4. Verified libraries for each database organization

Database organization	ACBLIB (ACBLIB data set is specified)	MODBLKS (MODBLKS data set is specified and DRD=NO is specified)	DFSMDA (DFSMDA data set is specified)	RECON (CHKRECON=YES is specified)	DRD (DRD=YES is specified)
HSAM (including SHSAM)	Yes	Yes	Yes	Yes	Yes
HISAM (including SHISAM)	Yes	Yes	Yes	Yes	Yes
HDAM	Yes	Yes	Yes	Yes	Yes
HIDAM	Yes	Yes	Yes	Yes	Yes

Table 4. Verified libraries for each database organization (continued)

Database organization	ACBLIB (ACBLIB data set is specified)	MODBLKS (MODBLKS data set is specified and DRD=NO is specified)	DFSMDA (DFSMDA data set is specified)	RECON (CHKRECON=YES is specified)	DRD (DRD=YES is specified)
INDEX (including Fast Path secondary indexes)	Yes	Yes	Yes	Yes	Yes
PHDAM	Yes	Yes	No (see note 1)	Yes	Yes
PHIDAM	Yes	Yes (see note 2)	No (see note 1)	Yes	Yes (see note 2)
PSINDEX	Yes	Yes (see note 2)	No (see note 1)	Yes	Yes (see note 2)
GSAM	No (see note 3)	No (see note 4)	No (see note 3)	No (see note 5)	No (see note 4)
LOGICAL	No (see note 3)	No (see note 4)	No (see note 6)	No (see note 6)	No (see note 4)
MSDB	Yes (see note 6)	Yes	No (see note 6)	No (see note 6)	Yes
DEDB	Yes	Yes	No (see note 7)	Yes	Yes

**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 (RDDs)

**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 RECONn data sets. The output consists of reports and an activity log.

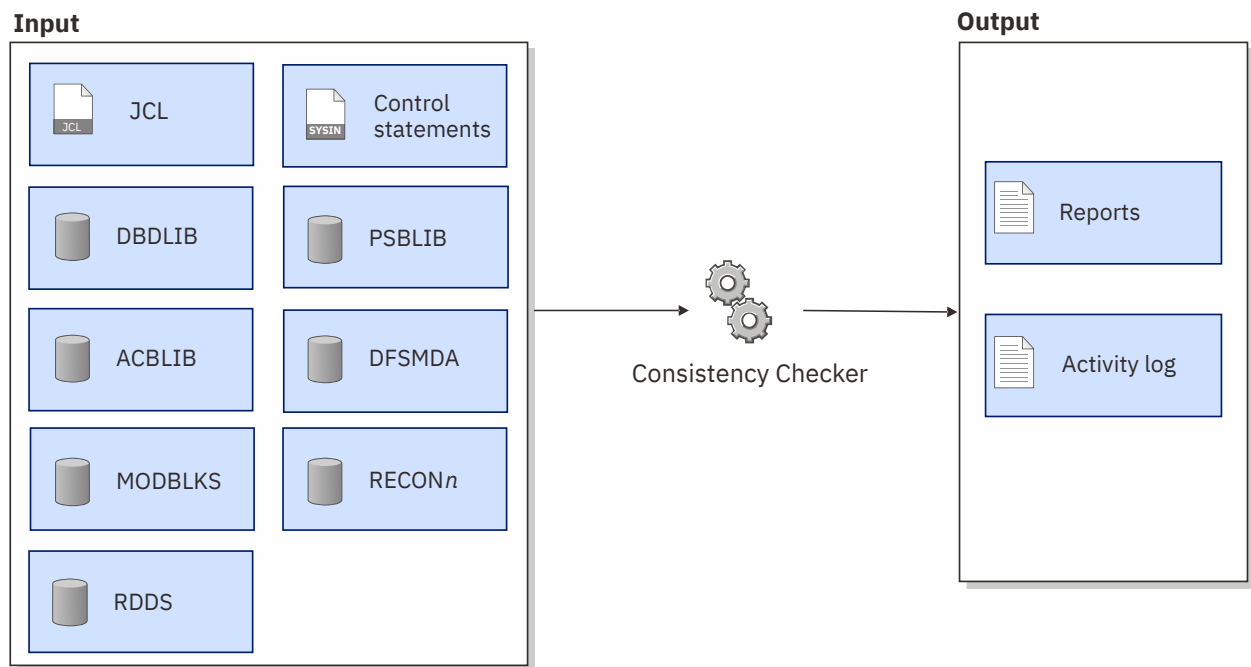


Figure 23. Data flow for Consistency Checker

## 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 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 116.
2. In the SYSIN data set, code the control statements for Consistency Checker.  
See [“Control statements for the Consistency Checker utility”](#) on page 118.
3. Submit the job.
4. Check the output data sets that are generated.  
See [“Output from the Consistency Checker utility”](#) on page 121.

### Related reference

[JCL examples for the Consistency Checker utility](#)

This topic provides JCL examples for running the Consistency Checker utility to check the consistency of DBDs and PSBs.

## 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” on page 116](#)
- [“EXEC statement” on page 116](#)
- [“DD statements” on page 116](#)

### 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
// DD DSN=IMSVS.SDFSRESL,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
//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](#).

#### **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 RECONn 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 RDDSs, 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 118 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”](#) on page 118
- [“Syntax rules”](#) on page 118
- [“Control statement keywords”](#) on page 118

### **Control statement example**

The following figure shows the control statements that can be coded in the SYSIN data set.

```
//SYSIN DD *
DDIRSFY=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:** DDIRSFY, PDIRSFY, CHKRECON, PCBERRMT, FAILONLY, and FAILRC statements can be specified only once, in no special order. DDIRSFY and CHKRECON are effective for all DBD members, PDIRSFY and



PCBERRLMT are effective for all PCB members, and FAILONLY and FAILRC are effective for all DBD and PSB members.

**DDIRSFX=[x|0]**

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=[x|0]**

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.

```
NOCHKORG=HSAM,GSAM
```

When specifying two or more statements, specify them as follows:

```
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

This topic provides 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.

```
//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
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMSVS.SDFSRESL,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.

```

//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
//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 *
        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.

```
FABL2010I CONTROL CARD SUPPLIED IS: DDIRSFX=A
FABL2010I CONTROL CARD SUPPLIED IS: PDIRSFX=A
FABL2010I CONTROL CARD SUPPLIED IS: FAILONLY=YES
FABL2010I CONTROL CARD SUPPLIED IS: FAILRC=04
FABL2010I CONTROL CARD SUPPLIED IS: CHKRECON=YES
FABL2010I CONTROL CARD SUPPLIED IS: DBD=TESTDB1
FABL2010I CONTROL CARD SUPPLIED IS: DBD=TESTDB2
FABL2010I CONTROL CARD SUPPLIED IS: PSB=TESTPSB1
FABL2010I CONTROL CARD SUPPLIED IS: PSB=TESTPSB2
FABL2007I PARAMETER USED IS: DDIRSFX=A
FABL2007I PARAMETER USED IS: PDIRSFX=A
FABL2007I PARAMETER USED IS: FAILONLY=YES
FABL2007I PARAMETER USED IS: FAILRC=04
FABL2007I PARAMETER USED IS: PCBERRLMT=2500
FABL2007I PARAMETER USED IS: CHKRECON=YES
FABL2006I ACBLIB DATA SET IS SPECIFIED
FABL2006I MODBLKS DATA SET IS SPECIFIED
FABL2006I DFSMDA DATA SET IS SPECIFIED
FABL2001I DBD TO BE PROCESSED IS TESTDB1
FABL2003E CONSISTENCY CHECK FAILED FOR TESTDB1
FABL2001I DBD TO BE PROCESSED IS TESTDB2
FABL2002I CONSISTENCY CHECK WAS SUCCESSFUL FOR TESTDB2
FABL2001I PSB TO BE PROCESSED IS TESTPSB1
FABL2003E CONSISTENCY CHECK FAILED FOR TESTPSB1
FABL2001I PSB TO BE PROCESSED IS TESTPSB2
FABL2002I CONSISTENCY CHECK WAS SUCCESSFUL FOR TESTPSB2
```

Figure 29. Messages 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” on page 122](#)
- [“Report field descriptions for the library information part” on page 123](#)
- [“Report field descriptions for the DB information part” on page 124](#)
- [“Report field descriptions for the DSG information part” on page 127](#)

### Sample report

The following figure shows an example of the DBD Check report.

**Note:** Even when the IMS installed level is updated to 15.2 and ACB, DBD, PSB, and RECON are generated with IMS, IMS 15.1 is still used for such resources.

LIBRARY INFORMATION

```

IMS       : 15.1
RECON    : 15.1
DFSRESLB : VOLUME=IMSVS  DSNAME=IMSVS.SDFSRESL
DBDLIB   : VOLUME=IMSVS  DSNAME=IMSVS.DBDLIB
ACBLIB   : VOLUME=IMSVS  DSNAME=IMSVS.ACBLIB
MODBLKS  : VOLUME=IMSVS  DSNAME=IMSVS.MODBLKS
DFSMDA   : VOLUME=IMSVS  DSNAME=IMSVS.MDALIB
RECON1   :                DSNAME=IMSVS.RECON1
RECON2   :                DSNAME=IMSVS.RECON2
RECON3   :                DSNAME=IMSVS.RECON3
    
```

SUFFIX = A

Figure 30. DBD Check report (Part 1 of 2)

DBD NAME : TESTDB1 VOLUME=IMSVS DSNAME=IMSVS.DBDLIB

DB INFORMATION

LIBRARY	CHK	ITEM/FIELD	CONTENTS	DBDLIB
ACBLIB		ACB MBR	FOUND	
		IMSREL	1510	1510
	***	GENDATE	04/15/2021 15:35	02/15/2021 11:25
		ACCESS	HDAM,OSAM	HDAM,OSAM
		SEGS	10	10
		RMNAME	RM@D01A	RM@D01A
		ANCH	1	1
		RBN	500	500
		BYTES	1024	1024
MODBLKS		DB DEF	FOUND	
		ACCSLVL	EXCLUSIVE	
DFSMDA		MDA MBR	FOUND	
RECON		DB RECORD	FOUND	
		SHRLVL	2	
		TYPE	IMS	
		DBORG/DSORG	HDAM,OSAM	
		BACKOT NEED	ON	
		PROHBT AUTH	OFF	
		RECOV NEED#	0	
		IC NEED#	6	

DSG INFORMATION

DSG #	CHK	D	DBDLIB	BLKSZ	ACBLIB	BLKSZ	M	DD1/DD2	DISP	DFSMDA / RECON	FLG	
1	1	D	DD@D01A	2048	DD@D01A	2048	B	DD@D01A	SHR	TESTDS.LIUUV2.TESTDB1.DD@D01A	I.-	
2	***	D	DD@D01BB	2048	DD@D01B	2048	M	DD@D01B	*	---	*	
3	1	D	DD@D01C	2048	DD@D01C	2048	B	DD@D01C	SHR	TESTDS.LIUUV2.TESTDB1.DD@D01C	I.-	
4	1	D	DD@D01D	2048	DD@D01D	2048	B	DD@D01D	SHR	TESTDS.LIUUV2.TESTDB1.DD@D01D	I.-	
5	1	D	DD@D01E	2048	DD@D01E	2048	B	DD@D01E	SHR	TESTDS.LIUUV2.TESTDB1.DD@D01E	I.-	
6	1	D	DD@D01F	2048	DD@D01F	2048	B	DD@D01F	SHR	TESTDS.LIUUV2.TESTDB1.DD@D01F	I.-	
-	***	D	---	---	---	---	M	DD@D01B	*	SHR	TESTDS.LIUUV2.TESTDB1.DD@D01B	*

LEGEND

MR - SHOWS 'DFSMDA / RECON' DATA IS:  
 B: RETRIEVED FROM BOTH DFSMDA AND RECON  
 M: RETRIEVED FROM DFSMDA  
 R: RETRIEVED FROM RECON

FLG - SHOWS RECON FLAGS ARE TURNED ON FOR:  
 I: IMAGE\_COPY\_NEEDED  
 R: RECOVERY\_NEEDED  
 P: PROHIBIT\_AUTHORIZATION

Figure 31. DBD Check report (Part 2 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

The utility, which can retrieve the IMS installed level of 15.2 or later, reports the IMS installed level instead of the IMS base level.

### 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

**ACBLIB**

ACB library specified in ACBLIB 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.

**RECON<sub>n</sub>**

DBRC RECON data set name

If CHKRECON=NO is specified in SYSIN DD, the lines for RECON<sub>n</sub> 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 113](#). 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:

**ACBLIB****ACB MBR**

Whether an ACB member corresponding to the DBD is found in the ACB library.

**GENDATE**

The date and time when the ACB was generated.

**Note:** If the generation date of the DBD is later than the generation date of the ACB, Consistency Checker regards them as inconsistent.

**IMSREL**

The IMS version and release that generated the ACB.

**Note:** If they are different from the IMS version and release defined in the DFSRESLB library, Consistency Checker regards them as inconsistent.

**ACCESS**

The DL/I access method and the operating system access method.

**SEGS**

The number of segments in the database. If it is a DEDB database, the number does not include the dummy segments.

**RMNAME**

The name of randomizing module. This field is shown only for an HDAM, a PHDAM, or a DEDB database.

**ANCH**

The number of root anchor points in each control interval or block. This field is shown only for an HDAM or a PHDAM database.

**RBN**

The maximum relative block number value. This field is shown only for an HDAM or a PHDAM database.

**BYTES**

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.

**XCI**

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.

**MODBLKS****DB DEF**

Whether a database definition that corresponds to the DBD is found in the specified MODBLKS module.

**ACCSLVL**

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.

### **ACCSLVL**

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.

## **NSYSRDDDS**

### **DB DEF**

Whether a database definition that corresponds to the DBD is found in the specified non-system RDDS.

### **ACCSLVL**

The access for the defined database:

**EX:**  
Exclusive

**UP:**  
Update

**RO:**  
Read only

**RD:**  
Read

NSYSRDDDS 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 SEGM 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” on page 129](#)
- [“Report field descriptions for the library information part” on page 129](#)
- [“Report field descriptions for the PSB information part” on page 130](#)
- [“Report field descriptions for the PCB information part” on page 133](#)

## Sample report

The following figure shows an example of the PSB Check report.

**Note:** Even when the IMS installed level is updated to 15.2 and ACB, DBD, PSB, and RECON are generated with IMS, IMS 15.1 is still used for such resources.

```

IMS LIBRARY INTEGRITY UTILITIES - CONSISTENCY CHECKER          "PSB CHECK REPORT"          PAGE: 1
5655-U08              DATE: 10/01/2021  TIME: 12.04.07        FABLECHK - V2.R2

LIBRARY INFORMATION
-----
IMS       : 15.1
DFSRESLB : VOLUME=IMSVS  DSNAME=IMSVS.SDFSRESL
PSBLIB   : VOLUME=IMSVS  DSNAME=IMSVS.PSBLIB
ACBLIB   : VOLUME=IMSVS  DSNAME=IMSVS.ACBLIB
MODBLKS  : VOLUME=IMSVS  DSNAME=IMSVS.MODBLKS

SUFFIX = A

```

Figure 32. PSB Check report (Part 1 of 2)

```

IMS LIBRARY INTEGRITY UTILITIES - CONSISTENCY CHECKER          "PSB CHECK REPORT"          PAGE: 2
5655-U08              DATE: 10/01/2021  TIME: 12.04.07        FABLECHK - V2.R2

PSB NAME : TESTPSB1  VOLUME=IMSVS  DSNAME=IMSVS.PSBLIB

PSB INFORMATION
-----
LIBRARY  CHK  ITEM/FIELD  CONTENTS  PSBLIB
-----
ACBLIB   *   ACB MBR      FOUND
          *   IMSREL      1510
          ***  GENDATE      05/20/2021 15:31  01/12/2021 11:14
          *   LANG        ASSEM/COBOL
          *   TTPC NO     3
          *   DBPCB NO    4
          *   MAXQ       1
          *   SSASIZE    3000
          *   IOEROPN    0
          *   LOCKMAX    0
          *   CMPAT      YES
          *   OLIC       NO
          *   GSROLBOK   NO
MODBLKS  *   PSB DEF     FOUND
          *   PGMTYPE    BATCH

TPPCB INFORMATION
-----
PCB #  CHK  P  LTERM/NAME  PCBNAME  ALTR  SAME  MODI  EXPR  LIST
-----
      2   P  PCBTTP1B  PCBTTP1B  NO    NO    YES  NO    YES
      *** A  TESTTP1B *  PCBTTP1B  NO    NO    NO  *  NO    YES

LEGEND
-----
PA - SHOWS 'PSBLIB / ACBLIB' DATA IS:
    B: RETRIEVED FROM BOTH PSBLIB AND ACBLIB
    P: RETRIEVED FROM PSBLIB
    A: RETRIEVED FROM ACBLIB

DBPCB INFORMATION
-----
PCB #  CHK  P  DBDNAME  PCBNAME  SENSEGS  KEYLN  PROCOPT  PROCSEQ(D)  VIEW  LIST
-----
      1   P  TESTDB1  PCBTDB1  10      24     G        TESTIDX1
      *** A  TESTDB1  PCBTDB11* 10      24     G        *
      4   P  TESTDB2  PCBTDB2  4        18     A
      *** A  TESTDB2  PCBTDB2  4        18     I        *

```

Figure 33. PSB Check report (Part 2 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 utility, which can retrieve the IMS installed level of 15.2 or later, reports the IMS installed level instead of the IMS base level.

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 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 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.

### **PGMTYPE**

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 RDDDS.

### **PGMTYPE**

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.

## **NSYSRDDDS**

### **PSB DEF**

Whether an application program definition that corresponds to the PSB is found in the specified non-system RDDDS.

### **PGMTYPE**

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.

NSYSRDDDS 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/ACBLIB

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 ends abnormally.

#### 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:

### PCB #

The sequential number of the DB 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/ACBLIB

These columns show that the following fields are found in the PSB and the ACB:

#### DBDNAME

DBD name.

#### PCBNAME (label)

The name of the PCB or the label for the PCB.

#### SENSEGS

The number of sensitive segments defined in the PCB.

#### KEYLN

The longest concatenated key length.

**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 135](#)
- [“Checking consistencies with the Multiple Resource Checker utility” on page 136](#)
- [“JCL requirements for the Multiple Resource Checker utility” on page 146](#)
- [“Control statements for the Multiple Resource Checker utility” on page 148](#)
- [“Fields compared in RECON data sets” on page 150](#)
- [“JCL examples for the Multiple Resource Checker” on page 155](#)
- [“Output from the Multiple Resource Checker utility” on page 159](#)

---

### 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.

Subsections:

- [“Function overview” on page 135](#)
- [“Program structure” on page 136](#)
- [“Restriction” on page 136](#)
- [“Data flow” on page 136](#)

### 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 150.

## 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.

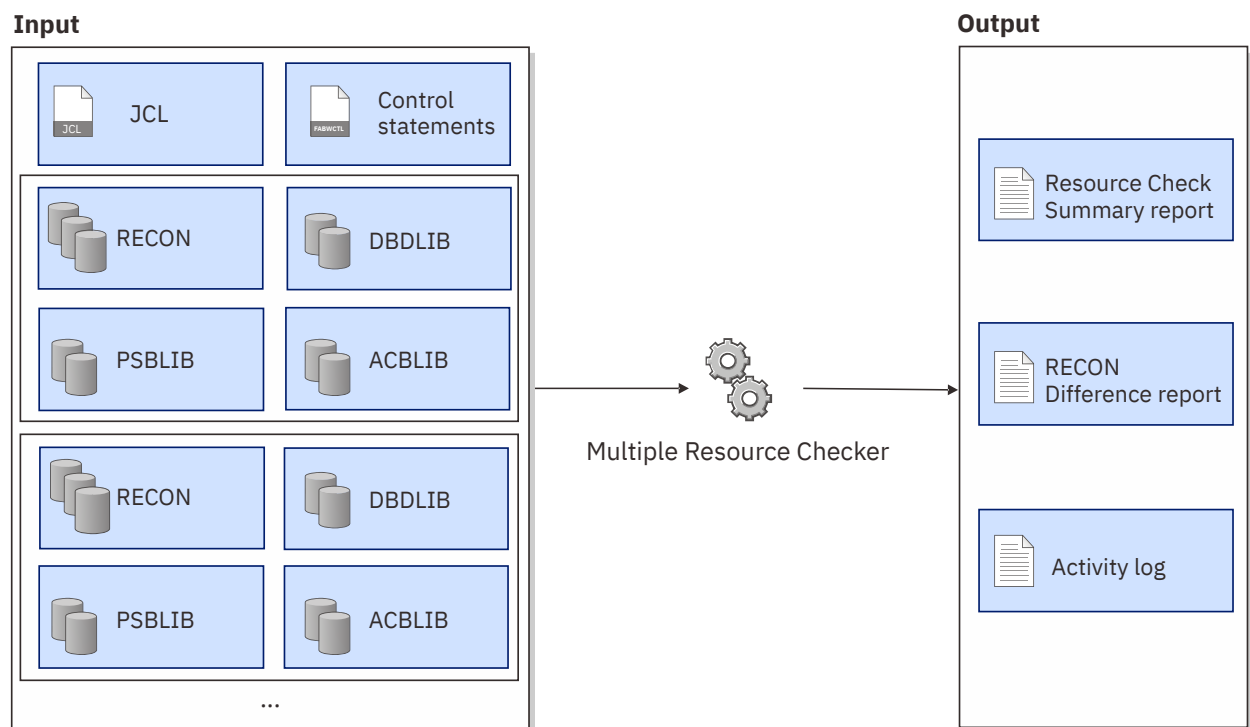


Figure 34. Data flow for Multiple Resource Checker

## 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.

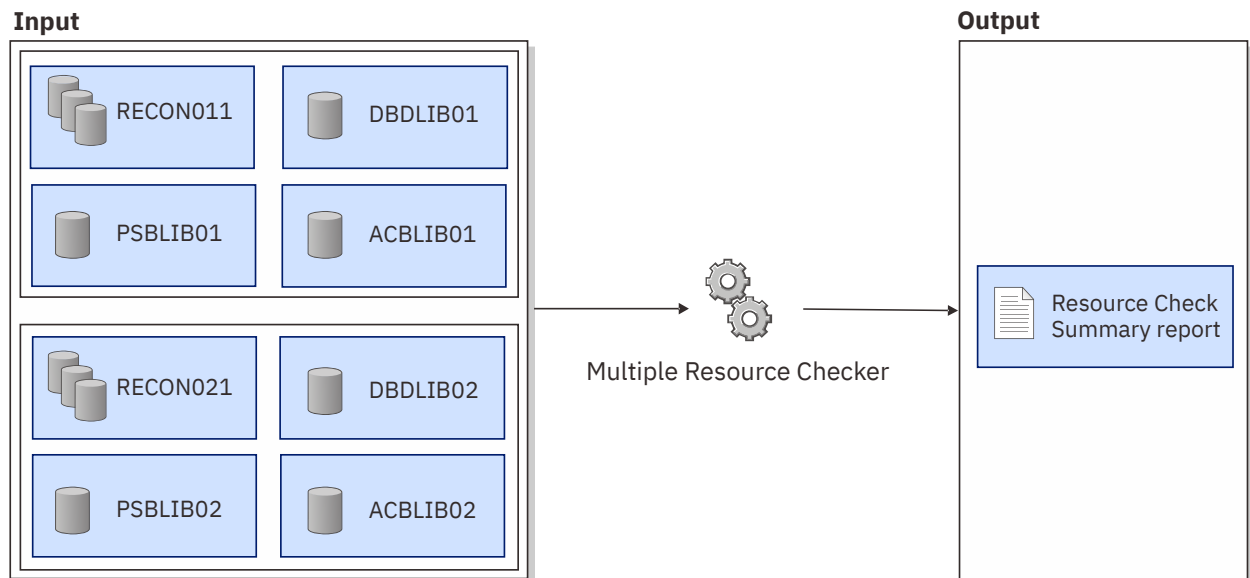


Figure 35. Checking the consistency of multiple resources

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:

```

//MULTIJB EXEC PGM=FABWMCHK
//STEPLIB DD DISP=SHR,DSN=HPS.SHPSLMD0
//IMSV14 DD DISP=SHR,DSN=IMS14A.SDFSRESL 1
//IMSV15 DD DISP=SHR,DSN=IMS15A.SDFSRESL
//RECON011 DD DISP=SHR,DSN=IMSVS.TEST.IMS14.RECON011 2
//RECON012 DD DISP=SHR,DSN=IMSVS.TEST.IMS14.RECON012
//RECON013 DD DISP=SHR,DSN=IMSVS.TEST.IMS14.RECON013
//RECON021 DD DISP=SHR,DSN=IMSVS.TEST.IMS15.RECON021
//RECON022 DD DISP=SHR,DSN=IMSVS.TEST.IMS15.RECON022
//RECON023 DD DISP=SHR,DSN=IMSVS.TEST.IMS15.RECON023
//DBDLIB01 DD DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB01 3
//DBDLIB02 DD DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB02
//PSBLIB01 DD DISP=SHR,DSN=IMSVS.TEST.IMS15.PSBLIB01
//PSBLIB02 DD DISP=SHR,DSN=IMSVS.TEST.IMS15.PSBLIB02
//ACBLIB01 DD DISP=SHR,DSN=IMSVS.TEST.IMS15.ACBLIB01
//ACBLIB02 DD DISP=SHR,DSN=IMSVS.TEST.IMS15.ACBLIB02
//FABWOUT DD SYSOUT=A 4
//FABWSUMM DD SYSOUT=A
//SYSPRINT DD SYSOUT=A 5

```

- **1** To check the RECON data sets generated by different IMS releases, code an IMSVnn DD statement for each IMS release.
- **2** To check two sets of RECON data sets, code three RECONxxn DD statements for each set of the RECON data sets.
- **3** To check DBDs, PSBs, and ACBs in different libraries, specify a DBDLIBxx, PSBLIBxx, ACBLIBxx DD statement for each library.
- **4** To generate messages and the Resource Check Summary report, specify the FABWOUT and FABWSUMM DD statements.
- **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 146.

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:

```

//FABWCTL DD *
NOCOMP=IMSREL
/*

```

For a description of the control statements, see [“Control statements for the Multiple Resource Checker utility”](#) on page 148.

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.

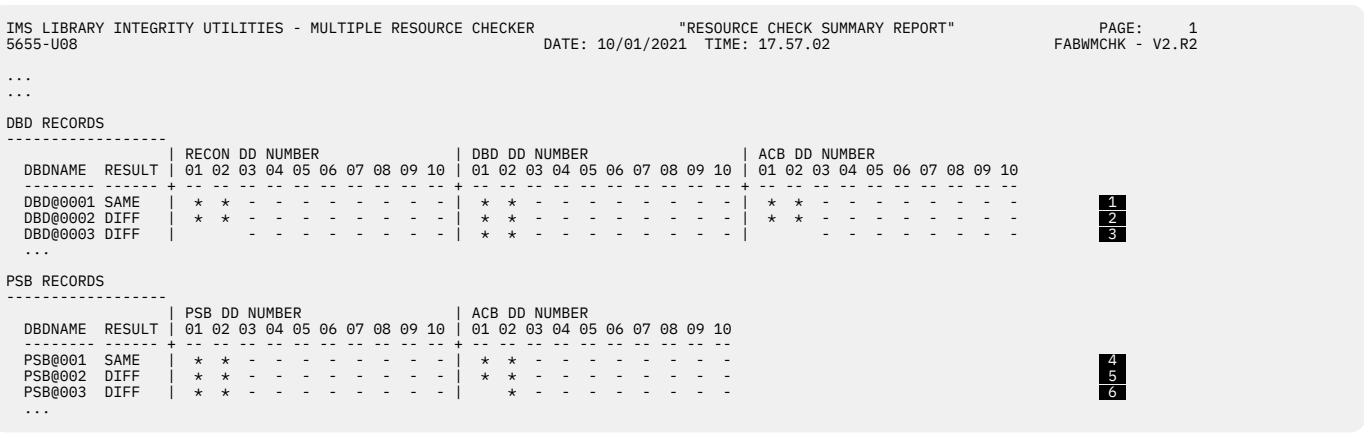


Figure 36. Report example: Resource Check Summary report for multiple resources

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 159](#).

## 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 167](#).
- 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 139](#).
- To view the database structure, use the DBD/PSB/ACB Mapper utility. See [Chapter 7, “DBD/PSB/ACB Mapper utility,” on page 201](#).

You can also use the DBD and PSB Map feature of IMS Administration Foundation to view database segment tree structures and program specifications through a web browser. For more information, see the *Unified Management Server User Guide*.

### Related reference

[Examples: Checking the consistency of multiple resources](#)

Use the examples in this topic to check the consistency of multiple resources with the Multiple Resource Checker utility.

## 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 150](#).

The following figure illustrates the resources that are compared in this scenario.

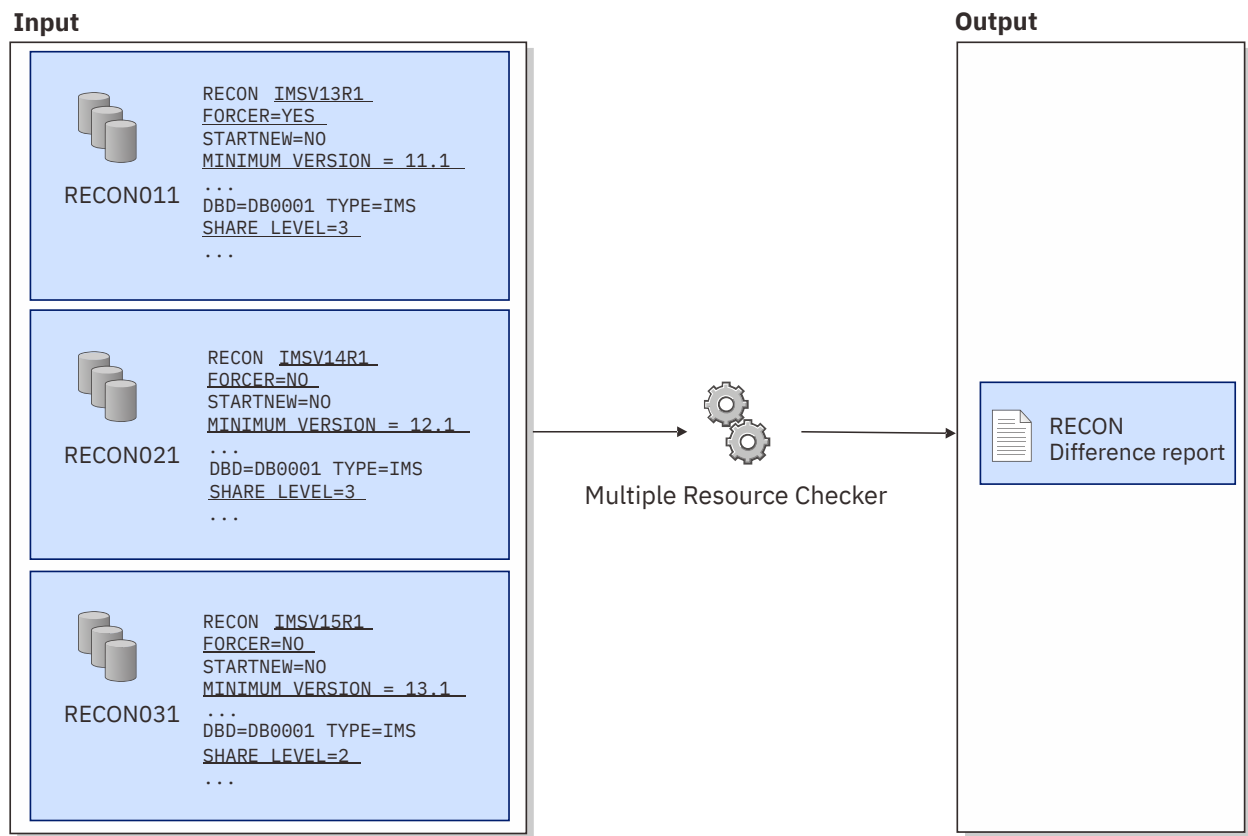


Figure 37. Checking the consistency of multiple sets of RECON data sets

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
//IMSV13 DD    DISP=SHR,DSN=IMS13A.SDFSRESL
//IMSV14 DD    DISP=SHR,DSN=IMS14A.SDFSRESL
//IMSV15 DD    DISP=SHR,DSN=IMS15A.SDFSRESL
//RECON011 DD  DISP=SHR,DSN=IMSVS.TEST.IMS13.RECON011
//RECON012 DD  DISP=SHR,DSN=IMSVS.TEST.IMS13.RECON012
//RECON013 DD  DISP=SHR,DSN=IMSVS.TEST.IMS13.RECON013
//RECON021 DD  DISP=SHR,DSN=IMSVS.TEST.IMS14.RECON021
//RECON022 DD  DISP=SHR,DSN=IMSVS.TEST.IMS14.RECON022
//RECON023 DD  DISP=SHR,DSN=IMSVS.TEST.IMS14.RECON023
//RECON031 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.RECON031
//RECON032 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.RECON032
//RECON033 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.RECON033
//FABWOUT DD   SYSOUT=A
//FABWSUMM DD  SYSOUT=A
//FABWRRPT DD  SYSOUT=A
//SYSPPRINT DD  SYSOUT=A
```

- **1** To check the RECON data sets generated by different IMS releases, code an IMSVnn DD statement for each IMS release.
- **2** To check three sets of RECON data sets, code three RECONxxn 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 FABWRRPT 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 146.

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 148.

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 FABWRRPT data set, also examine that report.

The following figure shows an example of the Resource Check Summary report.

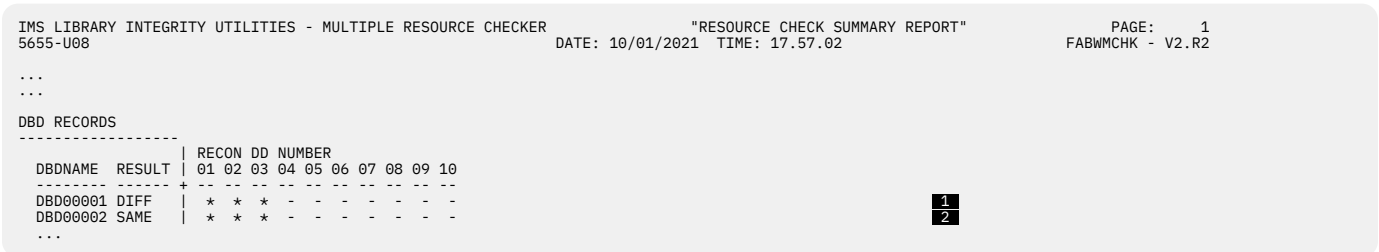


Figure 38. Report example: Resource Check Summary report for RECON data sets

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.

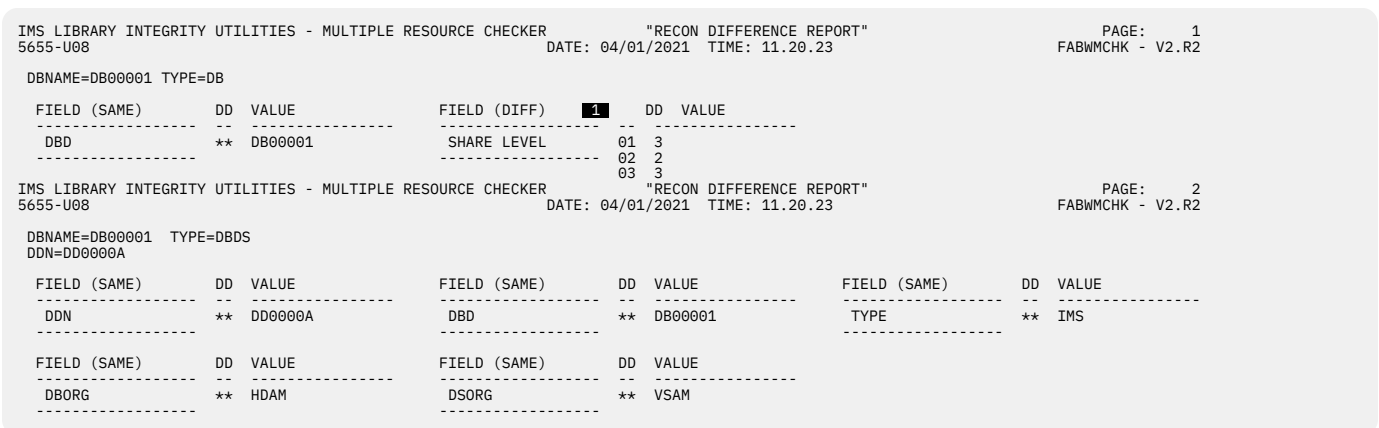


Figure 39. Report example: RECON Difference report when inconsistencies are detected

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 159](#)
- [“FABWRRPT data set” on page 162](#)

### Related reference

Example: [Comparing the database definitions across multiple sets of RECON data sets](#)

The JCL example in this topic is for generating a RECON Difference report, which reports details about the RECON fields that differ between multiple sets of RECON data sets.

## 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.

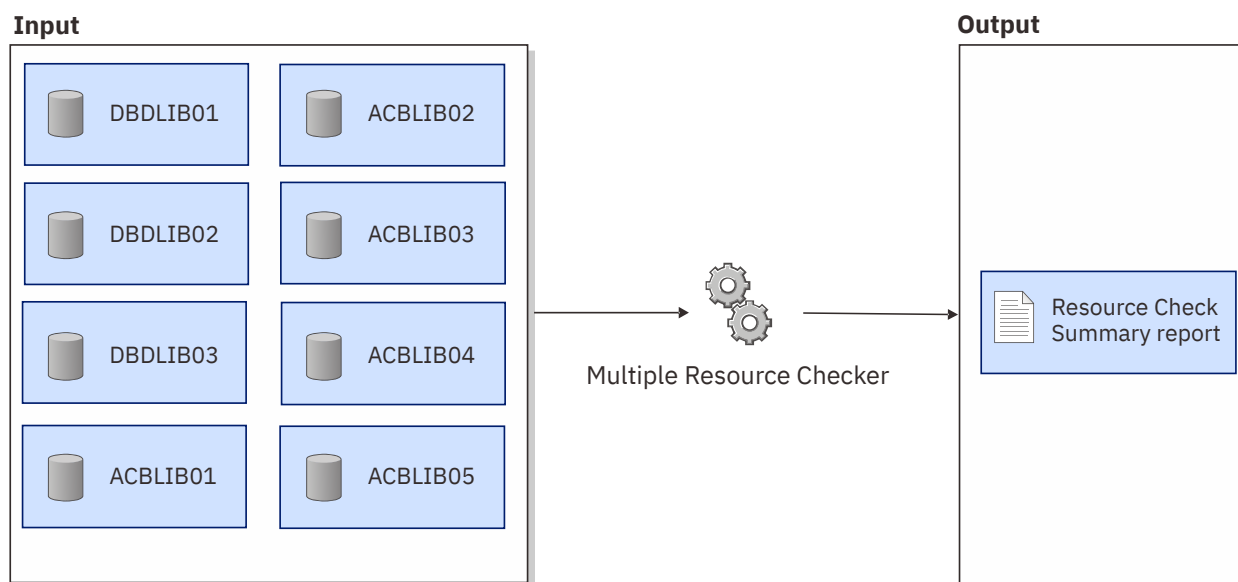


Figure 40. Checking the consistency of DBDs in DBD libraries and ACB libraries

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:



```

//MLTISTP      EXEC PGM=FABWMCHK
//STEPLIB DD   DISP=SHR,DSN=HPS.SHPSLMD0
//DBDLIB01 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB01
//DBDLIB02 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB02
//DBDLIB03 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB03
//ACBLIB01 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.ACBLIB01
//ACBLIB02 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.ACBLIB02
//ACBLIB03 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.ACBLIB03
//ACBLIB04 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.ACBLIB04
//ACBLIB05 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.ACBLIB05
//FABWOUT DD   SYSOUT=A
//FABWSUMM DD  SYSOUT=A

```

- **1** To check the DBD members in DBD libraries, specify a DBDLIBxx DD statement for each DBD library.
- **2** To check the DBD-type ACB members in ACB libraries, specify an ACBLIBxx DD statement for each ACB library.
- **3** 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 146.

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 148.

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.

IMS LIBRARY INTEGRITY UTILITIES - MULTIPLE RESOURCE CHECKER		"RESOURCE CHECK SUMMARY REPORT"										PAGE: 1									
5655-U08		DATE: 10/01/2021 TIME: 17.57.02										FABWMCHK - V2.R2									
DBD RECORDS																					
DBDNAME	RESULT	DBD DD NUMBER										ACB DD NUMBER									
		01	02	03	04	05	06	07	08	09	10	01	02	03	04	05	06	07	08	09	10
DBD00001	SAME	*	*	*	-	-	-	-	-	-	-	*	*	*	*	*	-	-	-	-	-
DBD00002	DIFF	*	*	*	-	-	-	-	-	-	-	*	*	*	*	*	-	-	-	-	-
DBD00003	DIFF	*	*	*	-	-	-	-	-	-	-	*	*	*	-	-	-	-	-	-	-

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 159.

## 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 167.
- To view the database structure, use the DBD/PSB/ACB Mapper utility. See [Chapter 7, “DBD/PSB/ACB Mapper utility,”](#) on page 201.

You can also use the DBD and PSB Map feature of IMS Administration Foundation to view database segment tree structures and program specifications through a web browser. For more information, see the *Unified Management Server User 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.

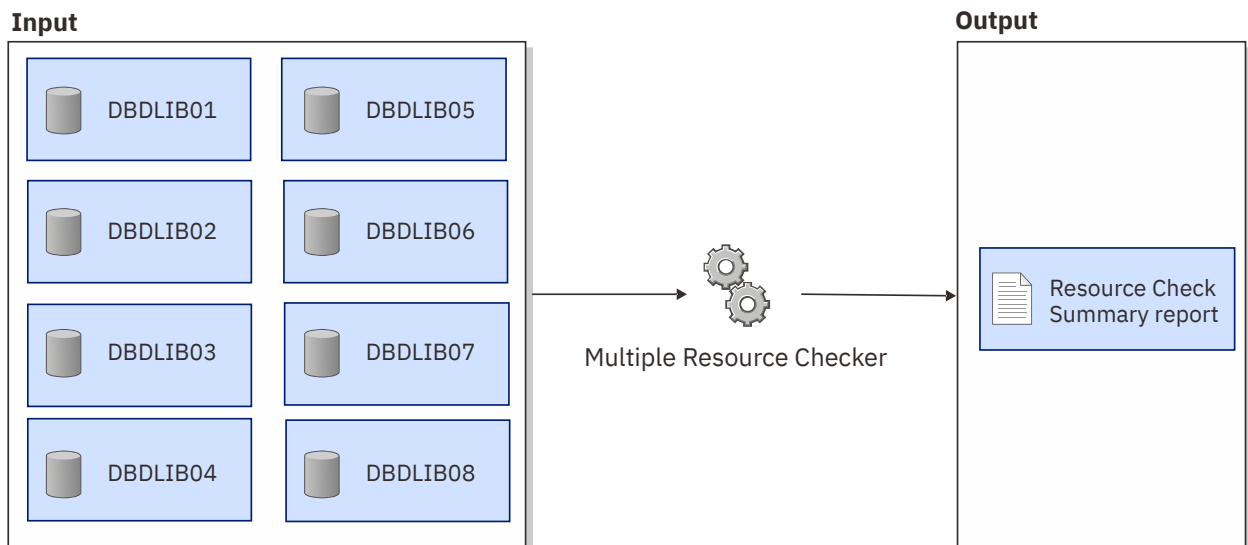


Figure 42. Checking the consistency of DBDs in multiple DBD libraries

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, DD statements, and control statements for the Multiple Resource Checker utility.

The following JCL example is for comparing DBD members in multiple DBD libraries:

```

//MLTISTP      EXEC PGM=FABWMCHK
//STEPLIB DD   DISP=SHR,DSN=HPS.SHPSLMD0
//DBDLIB01 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB01
//DBDLIB02 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB02
//DBDLIB03 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB03
//DBDLIB04 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB04
//DBDLIB05 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB05
//DBDLIB06 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB06
//DBDLIB07 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB07
//DBDLIB08 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.DBDLIB08
//ACBLIB01 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.ACBLIB01
//ACBLIB02 DD  DISP=SHR,DSN=IMSVS.TEST.IMS15.ACBLIB02
//FABWOUT DD   SYSOUT=A
//FABWSUMM DD  SYSOUT=A
//FABWCTL DD   *
      CHKONLY=DBD
/*

```

- **1** To check the DBD members in DBD libraries, specify a DBDLIBxx DD statement for each DBD library.
- **2** To generate messages and the Resource Check Summary report, specify the FABWOUT and FABWSUMM DD statements.
- **3** 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 146](#)
- [“Control statements for the Multiple Resource Checker utility” on page 148](#)

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.

IMS LIBRARY INTEGRITY UTILITIES - MULTIPLE RESOURCE CHECKER		"RESOURCE CHECK SUMMARY REPORT"		PAGE: 1							
5655-U08		DATE: 10/01/2021 TIME: 17.57.02		FABWMCHK - V2.R2							
... DBD RECORDS											
DBDNAME	RESULT	DBD DD NUMBER									
		01	02	03	04	05	06	07	08	09	10
DBD@0001	SAME	*	*	*	*	*	*	*	*	*	*
DBD@0002	DIFF	*	*	*	*	*	*	*	*	*	*
DBD@0003	DIFF	*	*	*	*	*	*	*	*	*	*
DBD@0004	DIFF	*	*	*	*	*	*	*	*	*	*
DBD@0005	DIFF	*	*	*	*	*	*	*	*	*	*
DBD@0006	DIFF	*	*	*	*	*	*	*	*	*	*
DBD@0007	SAME	*	*	*	*	*	*	*	*	*	*
DBD@0008	SAME	*	*	*	*	*	*	*	*	*	*
...											

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 159](#).

## 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 167.
- To view the database structure, use the DBD/PSB/ACB Mapper utility. See [Chapter 7, “DBD/PSB/ACB Mapper utility,”](#) on page 201.

You can also use the DBD and PSB Map feature of IMS Administration Foundation to view database segment tree structures and program specifications through a web browser. For more information, see the *Unified Management Server User 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”](#) on page 146
- [“EXEC statement”](#) on page 146
- [“DD statements”](#) on page 147

### 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.

```
//stepname EXEC PGM=FABWMCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//IMSV13 DD DSN=IMSV13.RESLIB,DISP=SHR
//IMSV14 DD DSN=IMSV14.RESLIB,DISP=SHR
//IMSV15 DD DSN=IMSV15.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:

```
//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 IMSPLEX 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=dbrcgrp'
```

### **IMSPLEX=*imsplex***

A 1-to-5 character IMSplex name to be used for RECON data sets.

### **DBRCGRP=*dbrcgrp***

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.

### **IMSV $nn$ 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 15, specify an IMSV15 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.

### **RECON $xxn$ 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.

#### **3**

Spare RECON data set.

### **DBDLIB $xx$ 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.

### **PSBLIB $xx$ 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.

### **ACBLIB $xx$ 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 148](#).

#### **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” on page 149](#)

- [“Syntax rules” on page 149](#)
- [“Control statement keywords” on page 149](#)

## 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 150.

**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 175.

**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

Field	CHKRECON= DBDEF	CHKRECON= DBDEF_RCV
DBD= <i>dbdname</i>	Compared	Compared
TYPE=IMS	Compared	Compared
SHARE LEVEL= <i>n</i>	Compared	Compared
DBRCVGRP= <i>rcvgrpnm</i>	-	Compared
FLAG: RECOVERABLE= YES   NO	-	Compared
FLAG: DATABASE LEVEL TRACK= YES   NO	-	Compared
IC NEEDED DISABLED	-	Compared

Table 6. DB (HALDB) record fields

Field	CHKRECON= DBDEF	CHKRECON= DBDEF_RCV
DBD= <i>dbdname</i>	Compared	Compared
TYPE=HALDB	Compared	Compared
SHARE LEVEL= <i>n</i>	Compared	Compared
DBRCVGRP= <i>rcvgrpnm</i>	-	Compared
PSNAME= <i>psname</i>	Compared	Compared
DBORG= <i>dbaseorg</i> DSORG= <i>dsetorg</i>	Compared	Compared
RECOVERABLE= YES   NO	-	Compared
PARTITIONS= <i>n</i>	Compared	Compared
ONLINE REORG CAPABLE= YES   NO	Compared	Compared
DATA SET GROUP MEMBERS	Compared	Compared
IC NEEDED DISABLED	-	Compared

Table 7. DB (PART) record fields

Field	CHKRECON= DBDEF	CHKRECON= DBDEF_RCV
DBD= <i>dbdname</i>	Compared	Compared
MASTER DB= <i>HALDB master name</i>	Compared	Compared
TYPE=PART	Compared	Compared

Table 7. DB (PART) record fields (continued)

Field	CHKRECON= DBDEF	CHKRECON= DBDEF_RCV
DSN PREFIX= <i>dsname</i>	-	Compared
PARTITION ID= <i>nnnnn</i>	Compared	Compared
REORG# = <i>nnnnn</i>	Compared	Compared
RANDOMIZER: NAME= ANCHOR= HIGH BLOCK#= BYTES= <i>nnnnnnnn</i>	Compared	Compared
FREE SPACE: FREE BLOCK FREQ FACTOR= <i>nnn</i> FREE SPACE PERCENTAGE= <i>nn</i>	Compared	Compared
PARTITION HIGH KEY/STRING (CHAR): (LENGTH=NNN)	Compared	Compared
PARTITION HIGH KEY/STRING (HEX):	Compared	Compared
OSAM BLOCK SIZE: <i>s</i> = <i>nnnnn</i>	Compared	Compared
ALTER BLOCK SIZE: <i>s</i> = <i>nnnnn</i>	Compared	Compared
DATABASE LEVEL TRACK=YES   NO	-	Compared
PARTITION DISABLED=YES   NO	-	Compared
ONLINE REORG CAPABLE= YES   NO	Compared	Compared

Table 8. DB (FP) record fields

Field	CHKRECON= DBDEF	CHKRECON= DBDEF_RCV
DBD= <i>dbdname</i>	Compared	Compared
TYPE=FP	Compared	Compared
SHARE LEVEL= <i>n</i>	Compared	Compared
RECOVERABLE= YES   NO	-	Compared
FULLSEG DEFAULT=YES   NO	Compared	Compared

Table 9. DBDS (non-FP) record fields

Field	CHKRECON= DBDEF	CHKRECON= DBDEF_RCV
DSN= <i>dsname</i>	-	Compared
TYPE=IMS	Compared	Compared
TYPE=PART	Compared	Compared
DBD= <i>dbdname</i> DDN= <i>ddname</i>	Compared	Compared
DBORG= <i>dbaseorg</i> DSORG= <i>dsetorg</i>	Compared	Compared
CAGRP= <i>cagrpname</i>	-	Compared
GENMAX= <i>nnnn</i>	-	Compared
REUSE   NOREUSE	-	Compared

Table 9. DBDS (non-FP) record fields (continued)

<b>Field</b>	<b>CHKRECON= DBDEF</b>	<b>CHKRECON= DBDEF_RCV</b>
RECOVPD= <i>nnn</i>	-	Compared
DEFLTJCL= <i>member</i>	-	Compared
ICJCL= <i>member</i>	-	Compared
OICJCL= <i>member</i>	-	Compared
RECOVJCL= <i>member</i>	-	Compared
RECVJCL= <i>member</i>	-	Compared

Table 10. DBDS (FP) record fields

<b>Field</b>	<b>CHKRECON= DBDEF</b>	<b>CHKRECON= DBDEF_RCV</b>
DBD= <i>dbdname</i> AREA= <i>areaname</i>	Compared	Compared
TYPE=FP	Compared	Compared
SHARE LEVEL= <i>n</i>	Compared	Compared
BORG= <i>dbaseorg</i> DSORG= <i>dsetorg</i>	Compared	Compared
CAGRP= <i>cagrpnam</i>	-	Compared
GENMAX= <i>nnnn</i>	-	Compared
REUSE NOREUSE	-	Compared
RECOVPD= <i>nnn</i>	-	Compared
VSO NOVSO	Compared	Compared
PREOPEN NOPREOPEN	Compared	Compared
PRELOAD NOPRELOAD	Compared	Compared
FULLSEG   NOFULLSG	Compared	Compared
CFSTR1= <i>cfstr_name</i>	Compared	Compared
CFSTR2= <i>cfstr_name</i>	Compared	Compared
LKASID   NOLKASID	Compared	Compared
MAS NOMAS	Compared	Compared
DEFLTJCL= <i>member</i>	-	Compared
ICJCL= <i>member</i>	-	Compared
RECVJCL= <i>member</i>	-	Compared
RECOVJCL= <i>member</i>	-	Compared
DBRCVGRP= <i>rcvgrpnm</i>	-	Compared
DATABASE LEVEL TRACK=YES NO	-	Compared

The fields in the following two tables are compared only when FABWRRPT DD is specified.

Table 11. RECON record fields

Field	CHKRECON= DBDEF	CHKRECON= DBDEF_RCV
RECOVERY CONTROL DATA SET, IMS VxRx	-	Compared
FORCER   NOFORCER	-	Compared
LOG DSN CHECK=xxxxxx	-	Compared
STARTNEW=YES NO	-	Compared
TRACEON   TRACEOFF	-	Compared
SSID=xxxxxxxx	-	Compared
LISTDLOG=YES NO	-	Compared
CA   IC   LOG DATA SETS CATALOGED=YES NO	-	Compared
LOG RETENTION PERIOD= <i>yy.ddd hh:mm:ss.t</i>	-	Compared
COMMAND AUTH=SAF   EXIT   BOTH   NONE	-	Compared
HLQ= <i>hql name</i>	-	Compared
RCNQUAL= <i>data_set_name</i>	-	Compared
CATALOG= <i>catalog_name</i>	-	Compared
ACCESS=SERIAL   PARALLEL	-	Compared
LIST=STATIC   CONCURR	-	Compared
SIZALERT=xxxxxxxx xxxxxxxx	-	Compared
LOGALERT=xxxxxxxx xxxxxxxx	-	Compared
REORG NUMBER VERIFICATION=YES   NO	-	Compared
IMSPLEX= <i>imsplex_name</i> , GROUP ID= <i>group_ID</i>	-	Compared

Table 12. Data group record fields

Field	CHKRECON= DBDEF	CHKRECON= DBDEF_RCV
GRPNAME= <i>grpname</i>	-	Compared
#MEMBERS= <i>nnn</i>	-	Compared
<i>dbdname ddname/areaname</i>	-	Compared

## JCL examples for the Multiple Resource Checker

The JCL examples in this topic help you understand how you can run the Multiple Resource Checker utility.

### Examples: Checking the consistency of multiple resources

Use the examples in this topic 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=FABWMCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//IMSV13 DD DSN=IMSV13.RESLIB,DISP=SHR
//IMSV14 DD DSN=IMSV14.RESLIB,DISP=SHR
//IMSV15 DD DSN=IMSV15.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=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
//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.

```

//stepname EXEC PGM=FABWMCHK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//IMSV13 DD DSN=IMSV13.RESLIB,DISP=SHR
//IMSV14 DD DSN=IMSV14.RESLIB,DISP=SHR
//IMSV15 DD DSN=IMSV15.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.

```

//stepname EXEC PGM=FABWMCHK
//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 JCL example in this topic 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
//IMSV13 DD DSN=IMSV13.RESLIB,DISP=SHR
//IMSV14 DD DSN=IMSV14.RESLIB,DISP=SHR
//IMSV15 DD DSN=IMSV15.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 FABWRRPT 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.

```

IMS LIBRARY INTEGRITY UTILITIES - MULTIPLE RESOURCE CHECKER          "MESSAGE"
5655-U08                      DATE: 10/01/2021  TIME: 17.57.02          PAGE:      1
                                          FABWMCHK - V2.R2
FABW0001I CONTROL STATEMENT SUPPLIED IS: DIFFREP=YES
FABW0001I CONTROL STATEMENT SUPPLIED IS: NOCHECK=IMSREL
FABW0001I CONTROL STATEMENT SUPPLIED IS: DBD=DBD@0001
FABW0001I CONTROL STATEMENT SUPPLIED IS: DBD=DBD@0002
FABW0002I PARAMETER USED IS: DIFFREP=NO
FABW0002I PARAMETER USED IS: NOCHECK=IMSREL

```

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” on page 160](#)
- [“Sample report” on page 161](#)

## 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 RECON $_{xxn}$  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 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 DBDLIB $_{xx}$ , PSBLIB $_{xx}$ , and ACBLIB $_{xx}$  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.

```

RECON DD NUMBER
-----
NUM DDNAME
-----
01 RECON011 : VOLUME=IMSVS1 DSNAME=IMSVS1.RECON1          IMS VER=V14R1 MINVERS=13.1 DB#=10
   RECON012 : VOLUME=IMSVS1 DSNAME=IMSVS1.RECON2
   RECON013 : VOLUME=IMSVS1 DSNAME=IMSVS1.RECON3
02 RECON021 : VOLUME=IMSVS1 DSNAME=IMSVS2.RECON1          IMS VER=V15R1 MINVERS=13.1 DB#=10
   RECON022 : VOLUME=IMSVS1 DSNAME=IMSVS2.RECON2
   RECON023 : VOLUME=IMSVS1 DSNAME=IMSVS2.RECON3

DBDLIB DD NUMBER
-----
NUM DDNAME
-----
01 DBDLIB01 : VOLUME=IMSVS1 DSNAME=IMSVS1.DBDLIB1
02 DBDLIB02 : VOLUME=IMSVS2 DSNAME=IMSVS2.DBDLIB1
03 DBDLIB03 : VOLUME=IMSVS3 DSNAME=IMSVS3.DBDLIB1

PSBLIB DD NUMBER
-----
NUM DDNAME
-----
01 PSBLIB01 : VOLUME=IMSVS1 DSNAME=IMSVS1.PSBLIB1
02 PSBLIB02 : VOLUME=IMSVS2 DSNAME=IMSVS2.PSBLIB1
03 PSBLIB03 : VOLUME=IMSVS3 DSNAME=IMSVS3.PSBLIB1
04 PSBLIB04 : VOLUME=IMSVS4 DSNAME=IMSVS4.PSBLIB1
05 PSBLIB05 : VOLUME=IMSVS5 DSNAME=IMSVS5.PSBLIB1
06 PSBLIB06 : VOLUME=IMSVS6 DSNAME=IMSVS6.PSBLIB1
07 PSBLIB07 : VOLUME=IMSVS7 DSNAME=IMSVS7.PSBLIB1
08 PSBLIB08 : VOLUME=IMSVS8 DSNAME=IMSVS8.PSBLIB1
09 PSBLIB09 : VOLUME=IMSVS9 DSNAME=IMSVS9.PSBLIB1
10 PSBLIB10 : VOLUME=IMSVSA DSNAME=IMSVSA.PSBLIB1

ACBLIB DD NUMBER
-----
NUM DDNAME
-----
01 ACBLIB01 : VOLUME=IMSVS1 DSNAME=IMSVS1.ACBLIB1
02 ACBLIB02 : VOLUME=IMSVS2 DSNAME=IMSVS2.ACBLIB1

DBD RECORDS
-----
DBDNAME  RESULT  RECON DD NUMBER          DBD DD NUMBER          ACB DD NUMBER
-----
01 02 03 04 05 06 07 08 09 10 01 02 03 04 05 06 07 08 09 10 01 02 03 04 05 06 07 08 09 10
DBD00001 SAME    * * - - - - - - - - * * * - - - - - * * - - - - - -
DBD00002 DIFF    * * - - - - - - - - * * * - - - - - * * - - - - - -
DBD00003 DIFF    * * - - - - - - - - * * * - - - - - * * - - - - - -
DBD00004 SAME    * * - - - - - - - - * * * - - - - - * * - - - - - -
DBD00005 SAME    * * - - - - - - - - * * * - - - - - * * - - - - - -
DBD00006 SAME    * * - - - - - - - - * * * - - - - - * * - - - - - -
DBD00007 SAME    * * - - - - - - - - * * * - - - - - * * - - - - - -
DBD00008 SAME    * * - - - - - - - - * * * - - - - - * * - - - - - -
DBD00009 DIFF    * - - - - - - - - * * * - - - - - * * - - - - - -
DBD00010 DIFF    * * - - - - - - - - * * * - - - - - * * - - - - - -

PSB RECORDS
-----
PSBNAME  RESULT  PSB DD NUMBER          ACB DD NUMBER
-----
01 02 03 04 05 06 07 08 09 10 01 02 03 04 05 06 07 08 09 10
PSB00001 SAME    * * * * * * * * * * * * - - - - - -
PSB00002 DIFF    * * * * * * * * * * * * - - - - - -
PSB00003 DIFF    * * * * * * * * * * * * - - - - - -
PSB00004 SAME    * * * * * * * * * * * * - - - - - -
PSB00005 SAME    * * * * * * * * * * * * - - - - - -

LEGEND
-----
SAME : THE COMPARED RESOURCES CONTAIN THE SAME INFORMATION.
DIFF : THE COMPARED RESOURCES CONTAIN DIFFERENT INFORMATION.
*    : INDICATES THAT THE MEMBER EXISTS.
(BLANK) : INDICATES THAT THE MEMBER DOES NOT EXIST.
-    : INDICATES THAT THE DD STATEMENT IS NOT SPECIFIED.

```

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” on page 163](#)
- [“Sample report” on page 163](#)

## Report field description

This report consists of the following fields.

### TYPE=

#### DBNAME=*dbname* TYPE=

TYPE= shows the type of the RECON record field. DBNAME= is printed before TYPE=.

TYPE= shows one of the following RECON fields:

#### RECON

RECON record fields

#### DBDSGRP

Data group record fields

#### DBGRP

Data group record fields

#### RECOVGRP

Data group record fields

#### DB

DB record fields

#### DBDS

DBDS record fields

For information about the RECON record fields, see the topic "Fields in a RECON listing, by record type" in *IMS Commands*.

### FIELD (SAME | DIFF)

The name of the RECON field and an indicator that shows whether the field values are the same.

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 150](#).

### 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.

TYPE=RECON

FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE
FORCER	**	NOFORCER	LOG DSN CHECK	**	CHECK17	STARTNEW	**	NO
GTF TRACE	**	TRACEOFF	SSID	**	**NULL**	LIST DLOG	**	NO
CA/IC/LOG CATALG	**	NO	LOGRET PERIOD	**	0.001 0:00:00.0	COMMAND AUTH	**	NONE
HLQ	**	**NULL**						
RCNQUAL	**	**NULL**						
CATALOG	**	**NULL**	ACCESS	**	SERIAL	LIST	**	STATIC
SIZALERT DSNUM	**	15	SIZALERT VOLNUM	**	16	SUZALERT PERCENT	**	95
LOGALERT DSNUM	**	3	LOGALERT VOLNUM	**	16	REORG# VERIFY	**	NO
IMSPLEX	**	** NONE **	GROUP ID	**	** NONE **			

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"
5655-U08                   DATE: 12/19/2021  TIME: 10.47.15          PAGE:      3
                                                                    FABWMCHK - V2.R2

DBNAME=DB00001 TYPE=DB

FIELD (SAME)  DD  VALUE          FIELD (SAME)  DD  VALUE          FIELD (SAME)  DD  VALUE
-----
DBD           **  DB00001          SHARE LEVEL   **  3              DBRCVGRP      **  **NULL**
-----

FIELD (SAME)  DD  VALUE          FIELD (SAME)  DD  VALUE          FIELD (DIFF)   DD  VALUE
-----
RECOVERABLE   **  NO              DB LEVEL TRACK **  N/A           IC NEEDED     01  DISABLED
                                                02  DISABLED
                                                03  ENABLED

IMS LIBRARY INTEGRITY UTILITIES - MULTIPLE RESOURCE CHECKER          "RECON DIFFERENCE REPORT"
5655-U08                   DATE: 12/19/2021  TIME: 10.47.15          PAGE:      4
                                                                    FABWMCHK - V2.R2

DBNAME=DB00001 TYPE=DBDS
DDN=DD0000A

FIELD (SAME)  DD  VALUE          FIELD (SAME)  DD  VALUE
-----
DDN           **  DD0000A          DBD           **  DB00001
-----

FIELD (SAME)  DD  VALUE
-----
DSN           **  IMSVS.TEST.IMS15.DD0000A
-----

FIELD (SAME)  DD  VALUE          FIELD (SAME)  DD  VALUE          FIELD (DIFF)   DD  VALUE
-----
TYPE          **  IMS              DBORG         **  HDAM           DSORG         01  VSAM
                                                02  VSAM
                                                03  OSAM

FIELD (DIFF)  DD  VALUE          FIELD (SAME)  DD  VALUE          FIELD (SAME)  DD  VALUE
-----
CAGRP        01  CAGRPD03          GENMAX        **  3              REUSE         **  NOREUSE
              02  CAGRPD03
              03  CAGR0003

FIELD (DIFF)  DD  VALUE          FIELD (SAME)  DD  VALUE          FIELD (SAME)  DD  VALUE
-----
RECOVPD      01  31              DEFLTJCL     **  **NULL**      ICJCL         **  ICJCL
              02  31
              03  365

FIELD (SAME)  DD  VALUE          FIELD (SAME)  DD  VALUE          FIELD (SAME)  DD  VALUE
-----
OICJCL       **  OICJCL          RECOVJCL     **  RECOVJCL      RECVJCL       **  ICRCVJCL
-----

```

Figure 56. Example of the RECON Difference report when CHKRECON=DBDEF\_RCV is specified (Part 2 of 3)

DBNAME=DB00002 TYPE=DB

FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE
DBD	**	DB00002	SHARE LEVEL	**	3	RECOVERABLE	**	YES

FIELD (DIFF)	DD	VALUE
FULLSEG DEFAULT	01	NOT SUPPORTED
	02	NO
	03	NO

DBNAME=DB00002 TYPE=DBDS  
 AREA=DD0002A

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 (DIFF)	DD	VALUE
DBORG	**	DEDB	DSORG	**	VSAM	CAGRP	01	GRP100
							02	GRP100
							03	GRP300

FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE	FIELD (DIFF)	DD	VALUE
GENMAX	**	3	REUSE	**	NOREUSE	RECOVPD	01	100
							02	100
							03	300

FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE
VSO	**	NOVSO	PREOPEN	**	NOPREOPEN	PRELOAD	**	NOPRELOAD

FIELD (DIFF)	DD	VALUE	FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE
FULLSEG	01	NOT SUPPORTED	CFSTR1	**	**NULL**	CFSTR2	**	**NULL**
	02	NOFULLSG						
	03	NOFULLSG						

FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE
LKASID	**	NOLKASID	MAS	**	NOMAS	DEFLTJCL	**	**NULL**

FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE
ICJCL	**	ICJCL	RECVJCL	**	ICRCVJCL	RECOVJCL	**	RECOVJCL

FIELD (SAME)	DD	VALUE	FIELD (SAME)	DD	VALUE
DBRCVGRP	**	**NULL**	DB LEVEL TRACK	**	YES

Figure 57. Example of the RECON Difference report when CHKRECON=DBDEF\_RCV is specified (Part 3 of 3)



---

## 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 167](#)
- [“Restrictions and considerations for the DBD/PSB/ACB Compare utility” on page 168](#)
- [“Comparing IMS control blocks” on page 169](#)
- [“JCL requirements for the DBD/PSB/ACB Compare utility” on page 170](#)
- [“Control statements for the DBD/PSB/ACB Compare utility” on page 171](#)
- [“JCL examples for the DBD/PSB/ACB Compare utility” on page 184](#)
- [“Output from the DBD/PSB/ACB Compare utility” on page 187](#)

---

### 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” on page 167](#)
- [“Program structure and job step” on page 167](#)
- [“Data flow” on page 168](#)

### 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.

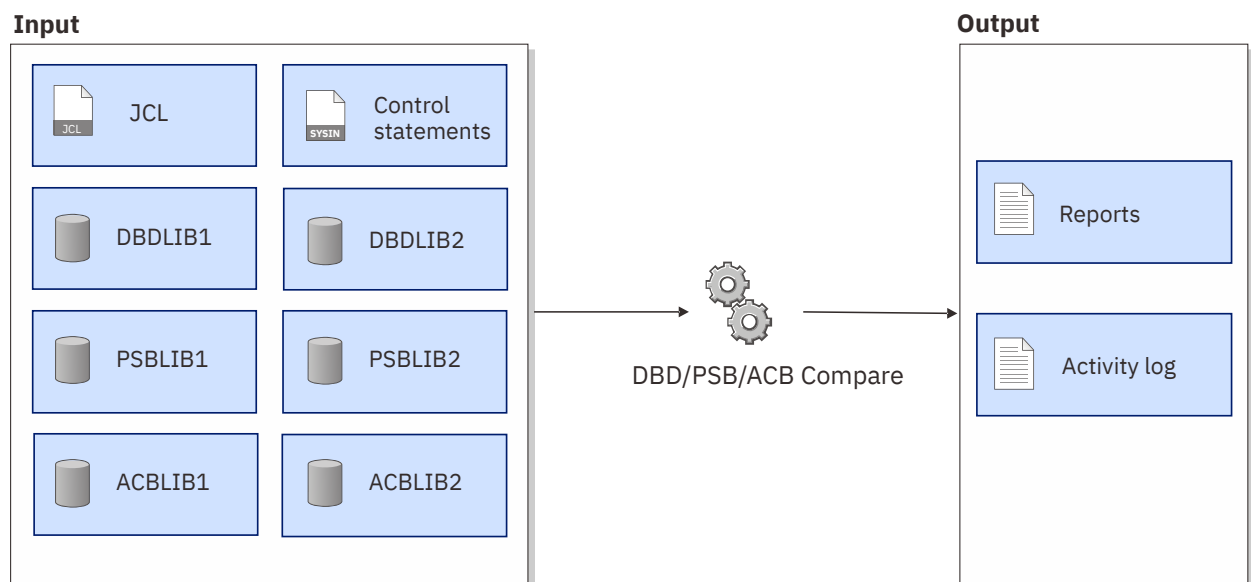


Figure 58. Data flow for DBD/PSB/ACB Compare

## 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 2.2, 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 ACB 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 245](#).
- 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 245](#).

## Comparing IMS control blocks

---

To compare DBDs, PSBs, and ACBs by using the DBD/PSB/ACB Compare utility, you must prepare 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 170](#).
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 171](#).
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 187](#).

### 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.

### Related reference

[JCL examples for the DBD/PSB/ACB Compare utility](#)

This topic provides JCL examples for running the DBD/PSB/ACB Compare utility to compare DBDs, PSBs, and ACBs.

## 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” on page 170](#)
- [“EXEC statement” on page 170](#)
- [“DD statements” on page 170](#)

### 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
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN 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 171.

## 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 172.

## REPORT control statement

A REPORT control statement controls report output. See [“REPORT control statement” on page 174](#).

## NOCOMP control statement

A NOCOMP control statement specifies the field that you want to exclude from comparison. See [“NOCOMP control statement” on page 175](#).

## 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” on page 172](#)
- [“PSB control statement” on page 173](#)
- [“ACB control statement” on page 173](#)
- [“Use of wildcards” on page 173](#)
- [“Quick reference for DBD, PSB, ACB control statements and DD statements” on page 174](#)

## 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=member1:member2**

The member (*member1*) in the library that is specified in the ACBLIB DD statement and the member (*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=member,parameter**

The members (*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 *parameter* (the allowed abbreviation is shown in parentheses):

**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.

**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.

**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.

**Use of wildcards**

For *member* and *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 *member2* (the second member name).

## 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.

Table 13. DBD/PSB/ACB Compare functions, control statements, and DD statements

Function	Control keyword	Required DD statements (O: Optional R: Required)								
		SYS PRINT	SYS OUT	DBD LIB	DBD LIB2	PSB LIB	PSB LIB2	ACB LIB	ACB LIB2	SYS IN
DBD compare	DBD=	R	R	R	R					R
	DBD=member1:member2	R	R	R	R					R
	DBD=member,A	R	R	R					R	R
PSB compare	PSB=	R	R			R	R			R
	PSB=member1:member2	R	R			R	R			R
	PSB=member,A	R	R			R			R	R
ACB compare	ACB=	R	R					R	R	R
	ACB=member1:member2	R	R					R	R	R
	ACB=member,D	R	R		R			R		R
	ACB=member,P	R	R				R	R		R
	ACB=member,B	R	R		R		R	R		R

## 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.

### NOREFPSB

This parameter specifies not to refer to a PSB-type ACB when the utility decodes an ACB for DEDB or MSDB to be compared in source-level comparison process. The utility obtains the DBD VERSION or EXIT parameter value of an ACB for a DEDB or MSDB from a PSB-type ACB that references the ACB. When the target library has many members, it can be time-consuming to obtain these values. You can specify this parameter to skip this process to obtain these values. This parameter is effective only when the SOURCE parameter is specified on the REPORT statement.

## Examples

You can code the REPORT control statement in the following ways:

- REPORT=SOURCE
- REPORT=CTLSTMT
- REPORT=SOURCE ,NODIFF
- REPORT=SOURCE ,NODIFF ,CTLSTMT ,NOREFPSB
- REPORT=SOURCE  
REPORT=NODIFF



REPORT=CTLSTMT  
 REPORT=NOREFPSB

## 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” on page 175](#)
- [“Summary of NOCOMP keyword parameters for block-level compare” on page 176](#)
- [“Description of NOCOMP keyword parameters” on page 182](#)
- [“Examples” on page 184](#)

### 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*

<b>NOCOMP keyword parameter</b>	<b>Affected member type</b>	<b>Statements and parameters not compared</b>
AREA	DBD	<b>AREA statement</b> AREA statements and any parameters on the AREA statements
COMPRTN	DBD	<b>SEGM statement</b> COMPRTN=
DBDNAME	DBD	<b>DBD statement</b> NAME=
IMSREL	DBD	<b>AREA, DATASET statements</b> DEVICE= (Removed by IMS 4. If a member is generated by IMS 3 or earlier and the other member is generated by IMS 4 or later, the utility does not compare this parameter.) <b>DBD statement</b> DATXEXIT= (Added by IMS 3. If a member is generated by IMS 2 or earlier and the other is generated by IMS 3 or later, the utility does not compare this parameter.)
	PSB	<b>PCB statement</b> PCBNAME=, LIST= (Added by IMS 3. If a member is generated by IMS 2 or earlier and the other is generated by IMS 3 or later, the utility does not compare these parameters.)
KEYLEN	PSB	<b>PCB statement</b> KEYLEN=
LANG	PSB	<b>PSBGEN statement</b> LANG=

Table 14. Source-level compare: Statements and parameters that are not compared (continued)

<b>NOCOMP keyword parameter</b>	<b>Affected member type</b>	<b>Statements and parameters not compared</b>
LIST	PSB	<b>PCB statement</b> LIST=
METADATA (or CATALOG)	DBD	<b>DFSMARSH, DFSMAP, DFSCASE statements</b> These statements and any parameters on these statements <b>FIELD statement</b> CASENAME=, DATATYPE=, DEPENDSON=, EXTERNALNAME=, MINOCCURS=, MAXOCCURS=, MAXBYTES=, PARENT=, REDEFINES=, RELSTART=, REMARKS=, STARTAFTER= <b>Other statements</b> ENCODING=, EXTERNALNAME=, REMARKS=
	PSB	EXTERNALNAME=, REMARKS=
PCBNAME	PSB	<b>PCB statement</b> PCBNAME= or label
PROCOPT	PSB	<b>DB PCB, GSAM PCB, SENSEG statements</b> PROCOPT=
PROCSEQ	PSB	<b>DB PCB statement</b> PROCSEQ=
PROCSEQD	PSB	<b>DB PCB statement</b> PROCSEQD=
PSB_ACCESS	PSB	<b>DB PCB statement</b> ACCESS=
PSB_PSELOPT	PSB	<b>DB PCB statement</b> PSELOPT=
PSBNAME	PSB	<b>PSBGEN statement</b> PSBNAME=
RMNAME	DBD	<b>DBD statement</b> RMNAME=
VERSION	DBD	<b>DBD statement</b> VERSION=  <b>Note:</b> NOCOMP=VERSION parameter specifies that the value of the VERSION= parameter of the DBD statement is not compared. It is not for the DBVER= parameter of the DBD statement, which is used for database versioning.

### 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 15. Block-level compare: Fields that are not compared

NOCOMP keyword parameter	Affected member type	Fields that are not compared (Section and field description)
AREA	DBD	Fields related to the AREA statement and the following fields: <b>IMS IDBD macro - PRFX DB section</b> <ul style="list-style-type: none"> <li>• NO OF SEGMENTS</li> </ul> <b>IMS IDBD macro - PRFX DSG section</b> <ul style="list-style-type: none"> <li>• LOGICAL RECORD LENGTH</li> <li>• OVERFLOW/OUTPUT LOGICAL RECORD LENGTH</li> </ul>
	DBD-type ACB	Fields related to the AREA statement and the following fields: <b>IMS DBFDMCB macro - DMCB DBD section</b> <ul style="list-style-type: none"> <li>• ADDRESS OF FDT ENTRY FOR ROOT KEY</li> <li>• MAXIMUM IOA LENGTH</li> <li>• OFFSET FROM BEGINNING OF DMCB</li> </ul>
COMPRTN	DBD	Fields related to the COMPRTN= parameter of the SEGM statement and the following fields: <b>IMS IDBD macro - PRFX DSG section</b> <ul style="list-style-type: none"> <li>• LONGEST SEGMENT LENGTH</li> </ul> <b>IMS IDBD macro - SEGTAB SEG section</b> <ul style="list-style-type: none"> <li>• MIN LENGTH OR ZERO FOR FIX LENGTH SEGS</li> </ul>
	DBD-type ACB	Fields related to the COMPRTN= parameter of the SEGM statement and the following fields: <b>IMS DFSDMB macro - PRFX section</b> <ul style="list-style-type: none"> <li>• DMB SIZE IN BYTES</li> </ul> <b>IMS DFSDMB macro - AMPBPX section</b> <ul style="list-style-type: none"> <li>• LENGTH OF LARGEST SEGMENT IN DATASET</li> </ul> <b>IMS DFSDMB macro - PSDB section</b> <ul style="list-style-type: none"> <li>• FOR VAR LENGTH SEG - MIN VALUE</li> </ul> <b>IMS DBFDMCB macro - DMCB DBD section</b> <ul style="list-style-type: none"> <li>• ADDRESS OF FDT ENTRY FOR ROOT KEY.</li> <li>• OFFSET FROM BEGINNING OF DMCB</li> </ul>
DBDNAME	DBD, DBD-type ACB	Fields related to the DBD NAME= statement
DMBNUM	DBD-type ACB	DMB number field

Table 15. Block-level compare: Fields that are not compared (continued)

NOCOMP keyword parameter	Affected member type	Fields that are not compared (Section and field description)
IMSREL	DBD	<p><b>IMS IDBD macro - DIR section</b></p> <ul style="list-style-type: none"> <li>• DBDGEN DONE ON IMSV12 OR LATER (Added by IMS 12)</li> </ul> <p><b>IMS IDBD macro - PRFX DB section</b></p> <ul style="list-style-type: none"> <li>• NO OF AREAS(NEW) (Added by IMS 8)</li> </ul> <p>If this field exists in both members, the utility compares this field.</p> <p><b>IMS IDBD macro - PRFX DSG section</b></p> <ul style="list-style-type: none"> <li>• AREA ID(NEW) (Added by IMS 8)</li> </ul> <p>If this field exists in both members, the utility compares this field.</p> <ul style="list-style-type: none"> <li>• DEVICE TYPE OR RESERVED FIELD (Removed by IMS 4)</li> </ul> <p><b>IMS IDBD macro - DBDXTB section</b></p> <ul style="list-style-type: none"> <li>• "DBDX" EYECATCHER (Added by IMS 3)</li> <li>• LENGTH OF DBDXTAB (Added by IMS 3)</li> <li>• LEVEL OF DBDGEN</li> <li>• CALL DFSDBUX1 USER EXIT</li> <li>• "V" = VERSION ID,</li> <li>• "T" = TIMESTAMP</li> <li>• LENGTH OF VERSION ID (HEX)</li> <li>• VERSION ID (VARIABLE LENGTH)</li> </ul> <p><b>IMS IDBD macro - EXITTB section</b></p> <ul style="list-style-type: none"> <li>• NODLET OPTION (Y/N)</li> <li>• NO BEFORE OPTION (Y/N)</li> </ul> <p><b>IMS DBDGEN macro - DBDGEN section</b></p> <ul style="list-style-type: none"> <li>• IMS RELEASE LEVEL</li> </ul>
	PSB	<p><b>IMS DFSIPSB macro - PRFX section</b></p> <ul style="list-style-type: none"> <li>• INCREASED LIMIT OF SENSEGS (Added by IMS 7)</li> <li>• IMS V12 FLAG (Added by IMS 12)</li> </ul> <p><b>IMS DFSIPSB macro - SENSEG section</b></p> <ul style="list-style-type: none"> <li>• PARENT OFFSET IN SEGTBL (Added by IMS 7)</li> </ul> <p><b>IMS PSBGEN macro - PSBGEN section</b></p> <ul style="list-style-type: none"> <li>• IMS RELEASE LEVEL (Added by IMS 3)</li> </ul>

Table 15. Block-level compare: Fields that are not compared (continued)

NOCOMP keyword parameter	Affected member type	Fields that are not compared (Section and field description)
KEYLEN	PSB	Fields related to the KEYLEN= parameter of the PCB statement
	PSB-type ACB	Fields related to the KEYLEN= parameter of the PCB statement and the following fields: <b>IMS DFSIPSB macro - PRFX section</b> <ul style="list-style-type: none"> <li>• PSB SIZE (W/O INTENT LIST)</li> <li>• PSB SIZE (WITH INTENT LIST)</li> <li>• SIZE OF CSA PART OF PSB</li> <li>• TOTAL SIZE OF PSB WORK AREA</li> <li>• MAX I/O WORK AREA SIZE</li> <li>• ADDR OF FAST PATH CONTROL BLOCK</li> <li>• LENGTH OF THIS DBPCB MINUS PREFIX</li> </ul>
LANG	PSB, PSB-type ACB	Field related to the LANG= parameter
LIST	PSB, PSB-type ACB	Field related to the LIST= parameter
METADATA (or CATALOG)	DBD, DBD-type ACB, PSB, PSB-type ACB	Fields related to the metadata fields in DBD, PSB, or ACB
PCBNAME	PSB	Fields related to the PCBNAME= parameter, the label parameter of the PCB statement, and the following fields: <b>IMS DFSIPSB macro - DBPCB PCB section</b> <ul style="list-style-type: none"> <li>• SEGMENT NAME FEEDBACK</li> </ul> <b>IMS DFSIPSB macro - GSPCB PCB section</b> <ul style="list-style-type: none"> <li>• SEGMENT NAME FEEDBACK</li> </ul>
	PSB-type ACB	Fields related to the PCBNAME= parameter, the label parameter of the PCB statement, and the following field: <b>IMS DFSIPSB macro - DBPCB PCB section</b> <ul style="list-style-type: none"> <li>• SEGMENT NAME FEEDBACK</li> </ul>

Table 15. Block-level compare: Fields that are not compared (continued)

NOCOMP keyword parameter	Affected member type	Fields that are not compared (Section and field description)
PROCOPT	PSB	Fields related to the PROCOPT= parameter
	PSB-type ACB	Fields related to the PROCOPT= parameter and the following fields: <b>IMS DFSIPSB macro - PPFX section</b> <ul style="list-style-type: none"> <li>• SIZE OF INDEX MAINT WORK AREA</li> <li>• SIZE OF SEGMENT WORK AREA</li> <li>• SIZE NEEDED FOR UPDATED DB LIST</li> <li>• SIZE OF SEGWK FOR GO EXPANSION</li> <li>• PSB SIZE (W/O INTENT LIST)</li> <li>• PSB SIZE (WITH INTENT LIST)</li> <li>• TOTAL SIZE OF PSB WORK AREA</li> <li>• MAX I/O WORK AREA SIZE</li> <li>• ADDR OF FAST PATH CONTROL BLOCK</li> </ul> <b>IMS DFSSDBM macro - SDB SEG (segname) section</b> <ul style="list-style-type: none"> <li>• RELATIVE OFFSET TO THE PHYSICAL DSG OF THIS SDB</li> <li>• ADDR OF THE DSG SECTION OF THE JCB FOR THIS SEG</li> </ul>
PROCSEQ	PSB	Fields related to the PROCSEQ= parameter
	PSB-type ACB	Fields related to the PROCSEQ= parameter and the following fields: <b>IMS DFSIPSB macro - PRFX section</b> <ul style="list-style-type: none"> <li>• SIZE OF INDEX MAINT WORK AREA</li> <li>• PSB SIZE (W/O INTENT LIST)</li> <li>• PSB SIZE (WITH INTENT LIST)</li> <li>• TOTAL SIZE OF PSB WORK AREA</li> <li>• MAX I/O WORK AREA SIZE</li> <li>• ADDR OF FAST PATH CONTROL BLOCK</li> </ul> <b>IMS DFSSDBM macro - SDB SEG (segname) section</b> <ul style="list-style-type: none"> <li>• SECONDARY INDEX IS MAIN PROCESSING SEQ.</li> <li>• RELATIVE OFFSET TO THE PHYSICAL DSG OF THIS SDB</li> <li>• ADDR OF THE DSG SECTION OF THE JCB FOR THIS SEG</li> <li>• SDB LOGICALLY RELATED</li> <li>• ADDRESS OF SDB EXPANSION</li> <li>• SEGMENT IS RETRIEVED VIA INDEX</li> <li>• SDB EXPANSION FOR SECONDAR IND</li> </ul>

Table 15. Block-level compare: Fields that are not compared (continued)

NOCOMP keyword parameter	Affected member type	Fields that are not compared (Section and field description)
PROCSEQD	PSB	Fields related to the PROCSEQD= parameter <b>IMS DFSIPSB macro - PRFX section</b> <ul style="list-style-type: none"> <li>• PST ADDR OF SCHED REGIN</li> <li>• EITHER 0000 OR THE OFFSET FROM</li> <li>• PSB SIZE</li> </ul>
	PSB-type ACB	Fields related to the PROCSEQD= parameter and the following fields: <b>IMS DFSIPSB macro - PRFX section</b> <ul style="list-style-type: none"> <li>• MAXIMUM DBPCB KEY FEEDBACK LENGTH</li> <li>• NUMBER OF DBPCBS IN THIS PSB</li> <li>• TOTAL NUMBER OF PCBS IN THIS PSB</li> <li>• PSB SIZE (W/O INTENT LIST)</li> <li>• PSB SIZE (WITH INTENT LIST)</li> <li>• SIZE OF CSA PART OF PSB</li> <li>• ADDR OF USER PARMS IN CTRL RGN</li> <li>• MAX I/O WORK AREA SIZE</li> <li>• ADDR OF FAST PATH CONTROL BLOCK</li> <li>• IMS/VIS DL/I DB ACCESS INDICATOR</li> <li>• OFFSET TO THE FIRST GSAM PCB</li> <li>• PSB SIZE</li> </ul>
PSB_ACCESS	PSB, PSB-type ACB	Fields related to the ACCESS= parameter
PSB_PSELOPT	PSB, PSB-type ACB	Fields related to the PSELOPT= parameter
PSBNAME	PSB, PSB-type ACB	Fields related to the PSBNAME= parameter
RMNAME	DBD	Fields related to the RMNAME= parameter
	DBD-type ACB	Fields related to the RMNAME= parameter and the following fields: <b>IMS DFSIPSB macro - PRFX section</b> <ul style="list-style-type: none"> <li>• DMB SIZE IN BYTES</li> <li>• ECB FOR BACKGROUND WRITE TO POST</li> </ul>
VERSION	DBD, DBD-type ACB, PSB-type ACB	Fields related to the VERSION= parameter <b>Note:</b> NOCOMP=VERSION parameter specifies that the value of the VERSION= parameter of the DBD statement is not compared. It is not for the DBVER= parameter of the DBD statement, which is used for database versioning.

## 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, DBD-type ACB, and PSB-type ACB members, the fields related to the VERSION= parameter of the DBDGEN utility is not compared.

**Note:** NOCOMP=VERSION parameter specifies that the value of the VERSION= parameter of the DBD statement is not compared. It is not for the DBVER= parameter of the DBD statement, which is used for database versioning.

## 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, PROCOPT, PROCSEQ  
NOCOMP=PROCSEQD, PSB_PSELOPT, PSB_ACCESS  
NOCOMP=METADATA
```

## JCL examples for the DBD/PSB/ACB Compare utility

This topic provides 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  
//SYSOUT DD SYSOUT=A  
//SYSPRINT DD SYSOUT=A  
//SYSIN DD *  
REPORT=SOURCE  
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  
//SYSOUT DD SYSOUT=A  
//SYSPRINT DD SYSOUT=A  
//SYSIN DD *  
REPORT=SOURCE  
DBD=BE1PARTS:BE2PARTS  
DBD=BE1ORDER:BE2ORDER  
DBD=BE1LORDR:BE2RORDR  
/*
```

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
//ACBLIB2 DD DSN=IMSVS.ACBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
REPORT=SOURCE
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
//PSBLIB2 DD DSN=IMSVS.TEST.PSBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
REPORT=SOURCE
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
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
REPORT=SOURCE
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
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
REPORT=SOURCE
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.

```
//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
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
REPORT=SOURCE
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.

```
//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
//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
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
REPORT=SOURCE
DBD=BE1PARTS
PSB=PE1CPINV
ACB=PE1CPINV
DBD=BE2LORDR
PSB=PE1CPPUR
ACB=PE1CPPUR
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.

```
IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB COMPARE          "MESSAGES"          PAGE :    1
5655-U08              DATE: 10/01/2021  TIME: 18.50.25          FABLCOMP - V2.R2
FABL0001I CONTROL CARD SUPPLIED IS:  REPORT=SOURCE
FABL0001I CONTROL CARD SUPPLIED IS:  DBD=TESTDB1
FABL0001I CONTROL CARD SUPPLIED IS:  PSB=TESTPSB1
FABL0001I CONTROL CARD SUPPLIED IS:  ACB=TESTDB1
FABL0001I CONTROL CARD SUPPLIED IS:  ACB=TESTPSB1
FABL0022I COMPARE MODE IS REPORT=SOURCE
FABL0004I DBD TO BE PROCESSED IS TESTDB1
FABL0007W DIFFERENCE FOUND DURING COMPARE DBD=TESTDB1
FABL0023I MEMBER TESTDB1 PROCESSED
FABL0004I PSB TO BE PROCESSED IS TESTPSB1
FABL0007W DIFFERENCE FOUND DURING COMPARE PSB=TESTPSB1
FABL0023I MEMBER TESTPSB1 PROCESSED
FABL0004I ACB TO BE PROCESSED IS TESTDB1
FABL0007W DIFFERENCE FOUND DURING COMPARE ACB=TESTDB1
FABL0023I MEMBER TESTDB1 PROCESSED
FABL0004I ACB TO BE PROCESSED IS TESTPSB1
FABL0006I NO DIFFERENCE FOUND DURING COMPARE ACB=TESTPSB1
FABL0023I MEMBER TESTPSB1 PROCESSED
FABL0042I COMPARED      1 DBD WITH DBD.  DETECTED      0 IDENTICAL SOURCES AND      1 MISMATCHED SOURCES.
FABL0042I COMPARED      1 PSB WITH PSB.  DETECTED      0 IDENTICAL SOURCES AND      1 MISMATCHED SOURCES.
FABL0042I COMPARED      2 ACB WITH ACB.  DETECTED      1 IDENTICAL SOURCES AND      1 MISMATCHED SOURCES.
```

Figure 69. Messages 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.

```
IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB COMPARE          "CONTROL STATEMENT REPORT"          PAGE: 1
5655-U08              DATE: 11/28/2021  TIME: 13.21.16          FABLCOMP - V2.R2

"CONTROL STATEMENTS"

0.....1.....2.....3.....4.....5.....6.....7.....8
1234567890123456789012345678901234567890123456789012345678901234567890

REPORT=SOURCE,CTLSTMT
NOCOMP=IMSREL,VERSION,DMBNUM,DBDNAME,PSBNAME,KEYLEN,PCBNAME,CATALOG,AREA
NOCOMP=RMNAME,COMPRTN,LANG,LIST,PROCOPT,PROCSEQ
NOCOMP=PROCSEQD,PSB_PSELOPT,PSB_ACCESS
DBD=DBD@D01A

"RUNTIME OPTIONS"
STATEMENT  PARAMETERS
-----
REPORT     SOURCE,CTLSTMT
NOCOMP     VERSION,DMBNUM,IMSREL,DBDNAME,PSBNAME,CATALOG,AREA,
           RMNAME,COMPRTN,PCBNAME,KEYLEN,LANG,LIST,PROCOPT,PROCSEQ,
           PSB_PSELOPT,PSB_ACCESS
```

Figure 70. 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” on page 188](#)
- [“Sample report: Block-level compare report for DBDs” on page 192](#)
- [“Sample report: Block-level compare report for PSBs” on page 193](#)
- [“Sample report: Block-level compare reports for ACBs” on page 194](#)

## 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

Section	Description
DIR	Information about the construction of the DBD control block
PRFX DB ( <i>dbname</i> )	Database information
PRFX DSG ( <i>ddname</i> )	Data set information
PRFX DSG ( <i>dsg-num</i> )	Data set information for HALDB
SEG TAB SEG ( <i>segname</i> )	Segment information
CMPRTN SEG ( <i>segname</i> ) CMP ( <i>exitname</i> )	Compression exit information
FLDTAB SEG ( <i>segname</i> ) FLD ( <i>fldname</i> )	Field information
LCHLD LCH ( <i>lchname</i> )	Logical child information
EXTDBD EXT ( <i>extdbnam</i> )	External DB information that is referred to by the DBD
INDXTB	Index information
SSPTAB	Subset pointer information about DEDB DBD
SORTAB SEG ( <i>segname</i> )	Source segment information
RDMRTN	HDAM randomizing routine information about HDAM DBD (see note)
DBDXTB	DBD extensional information
SEGXTB SEG ( <i>segname</i> )	Segment exit table information
EXITTB SEG ( <i>segname</i> ) EXT ( <i>exitname</i> )	Exit name array information
INDXMP PSL ( <i>exitname</i> )	FPSI Partition Selection exit information

Table 16. Sections in DBD Compare reports (continued)

Section	Description
DXVECT DXDENT DXDRET DXSENT SEG ( <i>segname</i> ) DXSEXT SEG ( <i>segname</i> ) DXSRET DXFMCT DXFDTT DXFDOT DXFXTT DXFRDT DXFRET DXFPAT DXFCIT DXFCAT DXFCRT DXFMDT DXFMNT DXFMRT DXFSAT DXMENT DXMPAT DXMOVT DXMITT DXMUTT DXMPOT DXMPOE DXMURT DXMRET DXLRET DXXRET DXTRET DXARET DXFEXT FLD ( <i>fldname</i> ) DXDRET	Metadata information for DBD
DBDGEN	IMS release level information. If the DBD was generated by IMS 3 or higher, the level is shown.
CKTBL1	/CK search field information about DEDB DBD
CKTBL2	/CK subsequence field information about DEDB DBD

**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

Section	Description
PRFX	PSB attributes information
TPPCB PCB ( <i>ltrimname</i> )	TP PCB information
DBPCB PCB ( <i>dbname</i> )	DB PCB information



Table 17. Sections in PSB Compare reports (continued)

<b>Section</b>	<b>Description</b>
GSPCB PCB ( <i>dbname</i> )	GSAM DB PCB information
SENSEG PCB ( <i>dbname</i> ) SEG ( <i>segname</i> )	Sensitive segment information about DB PCB
PSSPTB PCB ( <i>dbname</i> ) SEG ( <i>segname</i> )	Subset pointer information about DB PCB
SENFLD PCB ( <i>dbname</i> ) SEG ( <i>segname</i> ) FLD ( <i>fldname</i> )	Sensitive field information about DB PCB
REFTBL DB ( <i>dbname</i> )	DB information referred to by DB PCB or GSAM PCB
PCBNAM NAM ( <i>pcbname</i> )	PCB name information specified in the PCB statement
PXVECT PXPCRT PXPCXT PXSSRT PXSFR PXPSRT	Metadata information for PSB
PXXREF DB( <i>dbname</i> )	DB information referred to by DB PCB
PSBGEN	IMS release level information. If the PSB was generated by IMS 3 or higher, the level is shown.

Table 18. Sections in ACB Compare reports

<b>Section</b>	<b>Description</b>
PRFX	ACB attributes and database information
TPPCB PCB ( <i>ltrimname</i> )	TP PCB information
PCBNAM PCB ( <i>ltrimname</i> ) NAM ( <i>pcbname</i> )	PCB name information about TP PCB
DBPCB PCB ( <i>dbname</i> )	DB PCB information
DBPCBX PCB ( <i>dbname</i> )	DB PCB extensional information
VERID PCB ( <i>dbname</i> )	Version ID information about the DBD that is referred to by DB PCB
DMBXTB PCB ( <i>dbname</i> )	DMB extensional information about DEDB DMB referred to by DB PCB
SEGXTB PCB ( <i>dbname</i> ) SEG ( <i>segname</i> )	Segment exit table information about DEDB DMB referred to by DB PCB
EXITTB PCB ( <i>dbname</i> ) SEG ( <i>segname</i> ) EXT( <i>exitname</i> )	Exit name array information about DEDB DMB referred to by DB PCB
JCB PCB ( <i>pcbname</i> )	JOB control block information
SDB SEG ( <i>segname</i> )	Sensitive segment information about DB PCB
FSBLST	Information related to sensitive field of DB PCB
FSB FLD ( <i>fldname</i> )	Sensitive field information about DB PCB
SPCB	DB PCB information for DEDB DBD
SMLT SEG ( <i>segname</i> )	Sensitive segment information about DB PCB for DEDB DBD

Table 18. Sections in ACB Compare reports (continued)

Section	Description
RDMRTN	HDAM randomizing routine information about HDAM DBD
AMPBPX	Prefix information about the Access Method Prefix Block
AMPB	Access Method Prefix Block information
PSDB	Physical Segment Descriptor Block information
SECOND SEG ( <i>segname</i> )	Secondary list information
FDB FLD ( <i>fldname</i> )	Field Description Block information
DMBXTB	DMB extensional information
SEGXTB SEG ( <i>segname</i> )	Segment exit table information
EXITTB SEG ( <i>segname</i> ) EXT ( <i>exitname</i> )	Exit name array information
CPAC CMP ( <i>exitname</i> )	Compression exit information
DMCB DBD ( <i>dbdname</i> )	DEDB Master Control Block information about DEDB DBD
SDT SEG ( <i>segname</i> )	Segment information about DEDB DBD
FDT SEG ( <i>segname</i> ) FLD ( <i>fldname</i> )	Field information about DEDB DBD
MRMB DBD ( <i>dbdname</i> )	DEDB Randomizing Module Block information about DEDB DBD
DMAC DBD ( <i>dbdname</i> ) DSG ( <i>areaname</i> )	DEDB AREA Control List information about DEDB DBD
BHDR DBD ( <i>dbdname</i> ) FLD ( <i>fldname</i> )	Header information about MSDB DBD
CRTE	DEDB secondary index cross reference table entries information
CRTEXD	INDEX record partitioning information

ACB Compare reports also contain metadata information for DBDs and PSBs. For those sections, see [Table 16 on page 189](#) and [Table 17 on page 190](#).

#### 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.

TYPE : DBD  
NAME : DSFACHON  
LIBRARY 1 : VOLUME=IMSVS DSNAME=IMSVS.DBDLIB  
LIBRARY 2 : VOLUME=IMSVS DSNAME=IMSVS.TEST.DBDLIB

LIBRARY1	LIBRARY2	SECTIONS WHICH ARE DIFFERENT	DIFFERENCE DESCRIPTION
NO	YES	PRFX DB (DSFACHON)	HIDAM OSAM
YES	NO	PRFX DB (DSFACHON)	HIDAM VSAM
DSFACH00	DSFACH00	PRFX DSG (DSFACH00)	INPUT DD NAME/MSDB SEQUENCED FIELD NAME
3380	3350	PRFX DSG (DSFACH00)	DEVICE TYPE OR RESERVED FIELD
	57	PRFX DSG (DSFACH00)	LONGEST SEGMENT LENGTH
	2041	PRFX DSG (DSFACH00)	LOGICAL RECORD LENGTH
	2048	PRFX DSG (DSFACH00)	BLOCK/CI SIZE
	2041	PRFX DSG (DSFACH00)	OVERFLOW/OUTPUT LOGICAL RECORD LENGTH
	2048	PRFX DSG (DSFACH00)	OVERFLOW/OUTPUT BLOCK/CI SIZE
	35	SEGTAB SEG (SSFACP00)	DATA LEN - SEGM LEN FOR FIXED LEN SEGMS
ETEANA00	ETEANAME	FLDTAB SEG (SSFACP00) FLD (ETEANA00)	FIELD/XDFLD NAME
NO	YES	INDXTB	INDEX POINTER IS SYMBOLIC

Figure 71. DBD Compare report

The records in the figure have the following meanings:

**1st and 2nd records**

In the prefix section, different access methods are specified: HIDAM VSAM in LIBRARY 1, and HIDAM OSAM in LIBRARY 2.

**3rd record**

In the prefix section, different DD names are specified: DSFACH00 in LIBRARY 1, and DSFACH00 in LIBRARY 2.

**10th record**

In segment SSFACP00 in the SEGTAB section, different segment lengths are specified: 35 in LIBRARY 1, and 30 in LIBRARY 2.

**11th record**

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.

**Sample report: Block-level compare report for PSBs**

The following figure shows an example of the PSB Compare report.

TYPE : PSB  
NAME : PSBSMUUL  
LIBRARY 1 : VOLUME=IMSVS DSNAME=IMSVS.PSBLIB  
LIBRARY 2 : VOLUME=IMSVS DSNAME=IMSVS.TEST.PSBLIB

LIBRARY1	LIBRARY2	SECTIONS WHICH ARE DIFFERENT	DIFFERENCE DESCRIPTION
NO	YES	PRFX	APPLICATION PROGRAM IS ASSEMBLER, COBOL OR NOT SPECIFIED
YES	NO	PRFX	APPLICATION PROGRAM IS PL/I
GID	A	DBPCB PCB (DSFACHON)	PROCESSING OPTIONS
GID	A	SENSEG PCB (DSFACHON) SEG (SSFACP00)	SENSEG PROCOPT.
GID	A	SENSEG PCB (DSFACHON) SEG (SSFACP11)	SENSEG PROCOPT.
GID	A	SENSEG PCB (DSFACHON) SEG (SSFACP12)	SENSEG PROCOPT.
140	160	SENSEG PCB (DSSTUIVN) SEG (SSSTUP11)	PARENT OFFSET IN SEGTBL.
SSSTUP12		SENSEG PCB (DSSTUIVN) SEG (SSSTUP12)	SENSEG NAME.
A		SENSEG PCB (DSSTUIVN) SEG (SSSTUP12)	SENSEG PROCOPT.
140		SENSEG PCB (DSSTUIVN) SEG (SSSTUP12)	PARENT OFFSET IN SEGTBL.
		SENSEG PCB (DSSTUIVN) SEG (SSSTUP12)	SOURCE SEGMENT OFFSET.

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.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB COMPARE 5655-U08		"ACB COMPARE REPORT" DATE: 10/01/2021 TIME: 09.21.45		PAGE: 1 FABLACB0 - V2.R2
TYPE : ACB				
NAME : DSFACHON				
LIBRARY 1 : VOLUME=IMSVS DSNAME=IMSVS.ACBLIB				
LIBRARY 2 : VOLUME=IMSVS DSNAME=IMSVS.TEST.ACBLIB				
LIBRARY1	LIBRARY2	SECTIONS WHICH ARE DIFFERENT		DIFFERENCE DESCRIPTION
976	1088	PRFX		DMB SIZE IN BYTES
NO	YES	PRFX		HD INDEXED
YES	NO	PRFX		VSAM HIDAM
57	52	AMPBPX		LENGTH OF LARGEST SEGMENT IN DATASET
YES	NO	AMPB		ACCESS METHOD IS VSAM
YES	NO	AMPB		DATA SETS ARE PASSWORD PROTECTED
0	1690	AMPB		OVERFLOW BLOCKSIZE
DSFACH00	DSFACH00	AMPB		OVERFLOW DDNAME
35	30	PSDB		DATA LENGTH OF THE SEGMENT
35	30	PSDB		FOR VAR LENGTH SEG - MAX VALUE

Figure 73. 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.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB COMPARE 5655-U08		"ACB COMPARE REPORT" DATE: 10/01/2021 TIME: 09.21.45		PAGE: 1 FABLACB0 - V2.R2
TYPE : ACB				
NAME : PSBSMUUL				
LIBRARY 1 : VOLUME=IMSVS DSNAME=IMSVS.ACBLIB				
LIBRARY 2 : VOLUME=IMSVS DSNAME=IMSVS.TEST.ACBLIB				
LIBRARY1	LIBRARY2	SECTIONS WHICH ARE DIFFERENT		DIFFERENCE DESCRIPTION
NO	YES	PRFX		APPLICATION PROGRAM IS ASSEMBLER, COBOL OR NOT SPECIFIED
YES	NO	PRFX		APPLICATION PROGRAM IS PL/I
GID	A	DBPCB	PCB(DSSCHHVN)	PROCESSING OPTIONS
3	4	DBPCB	PCB(DSSCHHVN)	NO OF SENSITIVE SEGMENTS IN PCB
NO	YES	SDB	SEG(SSSCHP00)	SENSITIVITY IS REPLACE
000000F0	00000140	SDB	SEG(SSSCHP00)	SDB LOGICALLY RELATED
NO	YES	SDB	SEG(SSSCHP11)	SENSITIVITY IS REPLACE
00000140	00000190	SDB	SEG(SSSCHP11)	SDB LOGICALLY RELATED
SSSCHP13	SSSCHP12	SDB	SEG(SSSCHP13)	SEGMENT SYMBOLIC NAME
C8D7F1F3	C8D7F1F2	SDB	SEG(SSSCHP13)	NEXT SEGMENT ON LOGICAL TWIN CHAIN
NO	YES	SDB	SEG(SSSCHP13)	SENSITIVITY IS REPLACE
0	80	SDB	SEG(SSSCHP13)	OFFSET TO SIBLING OF SDB
00000190	000001E0	SDB	SEG(SSSCHP13)	SDB LOGICALLY RELATED
YES	NO	SDB	SEG(SSSCHP13)	SEGMENT HAS A PHYSICAL TWIN BKW
YES	NO	SDB	SEG(SSSCHP13)	SEGMENT HAS A LOGICAL TWIN BKWD
YES	NO	SDB	SEG(SSSCHP13)	SEGMENT HAS A LOGICAL PARENT PO
06	04	SDB	SEG(SSSCHP13)	POINTER NO IN PARENT TO FIRST OCCURANCE
00	05	SDB	SEG(SSSCHP13)	POINTER NO IN PARENT TO LAST OCCURANCE
13	02	SDB	SEG(SSSCHP13)	EXECUTABLE KEY LEN OF KEY FIELD
04	03	SDB	SEG(SSSCHP13)	SEGMENT CODE
GID	A	DBPCB	PCB(DSSTUIVN)	PROCESSING OPTIONS
NO	YES	SDB	SEG(SSSTUP00)	SENSITIVITY IS REPLACE
NO	YES	SDB	SEG(SSSTUP11)	SENSITIVITY IS REPLACE

Figure 74. Another 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” on page 195](#)
- [“Sample report: Source-level compare report for DBDs” on page 196](#)
- [“Sample report: Source-level compare report for PSBs” on page 196](#)
- [“Sample report: Source-level compare reports for ACBs” on page 197](#)

### 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.

**Note:** Even when the IMS installed level is updated to 15.2 and ACB, DBD, and PSB are generated with IMS, IMS 15.1 is still used for such resources.

#### 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.

```

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB COMPARE          "DBD COMPARE REPORT"
5655-U08                                                    DATE: 10/01/2021  TIME: 17.57.02
                                                                PAGE:          1
                                                                FABLDBD0 - V2.R2

TYPE       : DBD
NAME       : DBD@D03A
LIBRARY 1  : VOLUME=IMSVS  DSNAME=IMSVS.DBDLIB
LIBRARY 2  : VOLUME=IMSVS  DSNAME=IMSVS.TEST.DBDLIB

NUMBER OF DIFFERENT STATEMENTS
INSERTED   : 1
DELETED    : 0
CHANGED    : 3

      DBDGEN: 08/04/2021 17.41  | DBDGEN: 08/04/2021 17.41
      IMSREL: 1510              | IMSREL: 1510

CHK LIBRARY 1 SOURCE LINES  | LIBRARY 2 SOURCE LINES
-----1-----2-----3--  | -----1-----2-----3--
C - DBD                      | DBD
*   NAME=DBD@D03A,          |   NAME=DBD@D03A,
   ACCESS=(HDAM,OSAM),     |   ACCESS=(HDAM,VSAM),
   RMNAME=(RNM,2,500,800), |   RMNAME=(RNM,2,500,800),
   PASSWD=NO,              |   PASSWD=NO,
   VERSION= 08/04/21 17.41 |   VERSION= 08/04/21 17.41
C - DSG1 DATASET           | DSG1 DATASET
*   DD1=DD@D03A,          |   DD1=DD@D03A,
   SIZE=(1690),           |   SIZE=(2048),
   SCAN=3,                |   SCAN=3,
   FRSPC=(2,3)            |   FRSPC=(2,3)
C - SEGM                   | SEGM
*   NAME=D03SEG1,         |   NAME=D03SEG1,
   PARENT=0,              |   PARENT=0,
   BYTES=100,             |   BYTES=105,
   RULES=(LLL, LAST),    |   RULES=(LLL, LAST),
   PTR=(TWIN,,,,)        |   PTR=(TWINBWD,,,,)
FIELD                     | FIELD
NAME=(D03FLD1A,SEQ,U),   | NAME=(D03FLD1A,SEQ,U),
START=1,                 | START=1,
BYTES=10,                | BYTES=10,
TYPE=C                   | TYPE=C
I -                         | FIELD
                             | NAME=(D01FLD1B),
                             | START=11,
                             | BYTES=10,
                             | TYPE=X
SEGMENT                   | SEGM
NAME=D03SEG2,           | NAME=D03SEG2,
PARENT=((D03SEG1,)),    | PARENT=((D03SEG1,)),
BYTES=100,              | BYTES=100,
RULES=(LLL, LAST),     | RULES=(LLL, LAST),
PTR=(TWIN,,,,)         | PTR=(TWIN,,,,)
FIELD                     | FIELD
NAME=(D03FLD2A,SEQ,U), | NAME=(D03FLD2A,SEQ,U),
START=1,                 | START=1,
BYTES=10,                | BYTES=10,
TYPE=C                   | TYPE=C
DBDGEN                   | DBDGEN
FINISH                   | FINISH
END                       | END

```

Figure 75. DBD source-level compare report

**Sample report: Source-level compare report for PSBs**

The following figure is an example of a PSB Compare report.

TYPE : PSB  
NAME : PSB@003  
LIBRARY 1 : VOLUME=IMSVS DSNAME=IMSVS.PSBLIB  
LIBRARY 2 : VOLUME=IMSVS DSNAME=IMSVS.TEST.PSBLIB

NUMBER OF DIFFERENT STATEMENTS  
INSERTED : 0  
DELETED : 2  
CHANGED : 1

PSBGEN: 07/20/2021 14.33 | PSBGEN: 07/20/2021 14.34  
IMSREL: 1510 | IMSREL: 1510

CHK	LIBRARY 1 SOURCE LINES	LIBRARY 2 SOURCE LINES
	-----1-----2-----3--	-----1-----2-----3--
	PCB	PCB
	TYPE=DB,	TYPE=DB,
	DBDNAME=DBD@S01A,	DBDNAME=DBD@S01A,
	PROCOPT=G,	PROCOPT=G,
	KEYLEN=30	KEYLEN=30
	SENSEG	SENSEG
	NAME=S02SEG1,	NAME=S02SEG1,
	PARENT=0	PARENT=0
C -	PCB	PCB
*	TYPE=DB,	TYPE=DB,
	DBDNAME=DBD@M02A,	DBDNAME=DBD@M01A,
	PROCOPT=G,	PROCOPT=G,
	KEYLEN=30	KEYLEN=30
	SENSEG	SENSEG
	NAME=M02SEG1,	NAME=M02SEG1,
	PARENT=0	PARENT=0
D -	PCB	
	TYPE=DB,	
	DBDNAME=DBD@S03A,	
	PROCOPT=G,	
	KEYLEN=30	
D -	SENSEG	
	NAME=S03SEG1,	
	PARENT=0	
	PSBGEN	PSBGEN
	PSBNAME=PSB@003,	PSBNAME=PSB@003,
	LANG=PL/I,	LANG=PL/I,
	IOEROPN=(100,WTOR),	IOEROPN=(100,WTOR),
	MAXQ=10,	MAXQ=10,
	OLIC=YES,	OLIC=YES,
	CMPAT=YES	CMPAT=YES
	END	END

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.

TYPE : ACB(DBD)  
NAME : DBD@D03A  
LIBRARY 1 : VOLUME=IMSVS DSNAME=IMSVS.ACBLIB  
LIBRARY 2 : VOLUME=IMSVS DSNAME=IMSVS.TEST.ACBLIB

NUMBER OF DIFFERENT STATEMENTS  
INSERTED : 1  
DELETED : 0  
CHANGED : 3

ACBGEN: 07/20/2021 14.48 | ACBGEN: 07/20/2021 14.48  
IMSREL: 1510 | IMSREL: 1510

CHK	LIBRARY 1 SOURCE LINES	LIBRARY 2 SOURCE LINES
	1-----2-----3--	1-----2-----3--
C -	DBD	DBD
	NAME=DBD@D03A,	NAME=DBD@D03A,
	ACCESS=(HDAM,VSAM),	ACCESS=(HDAM,VSAM),
*	RMNAME=(RNM,2,500,800),	RMNAME=(RNM,5,500,800),
	PASSWD=NO,	PASSWD=NO,
	VERSION= 07/20/21 14.48	VERSION= 07/20/21 14.48
	DSG1 DATASET	DSG1 DATASET
	DD1=DD@D03A,	DD1=DD@D03A,
	SCAN=3,	SCAN=3,
	FRSPC=(2,3)	FRSPC=(2,3)
C -	SEGM	SEGM
*	NAME=D03SEG1,	NAME=D01SEG1,
	PARENT=0,	PARENT=0,
*	BYTES=100,	BYTES=105,
	RULES=(LLL, LAST),	RULES=(LLL, LAST),
	PTR=(TWIN, , ,)	PTR=(TWIN, , ,)
	FIELD	FIELD
	NAME=(D03FLD1A, SEQ, U),	NAME=(D03FLD1A, SEQ, U),
	START=1,	START=1,
	BYTES=10,	BYTES=10,
	TYPE=C	TYPE=C
I -		FIELD
		NAME=(D01FLD1B),
		START=11,
		BYTES=10,
		TYPE=X
C -	SEGM	SEGM
	NAME=D03SEG2,	NAME=D03SEG2,
*	PARENT=((D03SEG1,)),	PARENT=((D01SEG1,)),
	BYTES=100,	BYTES=100,
	RULES=(LLL, LAST),	RULES=(LLL, LAST),
	PTR=(TWIN, , ,)	PTR=(TWIN, , ,)
	FIELD	FIELD
	NAME=(D03FLD2A, SEQ, U),	NAME=(D03FLD2A, SEQ, U),
	START=1,	START=1,
	BYTES=10,	BYTES=10,
	TYPE=C	TYPE=C
	DBDGEN	DBDGEN
	FINISH	FINISH
	END	END

Figure 77. ACB source-level compare report (Sample 1)

The following figure is another example of an ACB Compare report.



TYPE : ACB(PSB)  
NAME : PSB@003  
LIBRARY 1 : VOLUME=IMSVS DSNAME=IMSVS.ACBLIB  
LIBRARY 2 : VOLUME=IMSVS DSNAME=IMSVS.TEST.ACBLIB

NUMBER OF DIFFERENT STATEMENTS  
INSERTED : 0  
DELETED : 2  
CHANGED : 2

ACBGEN: 07/20/2021 14.48 | ACBGEN: 07/20/2021 14.48  
IMSREL: 1510 | IMSREL: 1510

CHK	LIBRARY 1 SOURCE LINES	LIBRARY 2 SOURCE LINES
	1-----2-----3--	1-----2-----3--
	PCB	PCB
	TYPE=DB,	TYPE=DB,
	DBDNAME=DBD@S01A,	DBDNAME=DBD@S01A,
	PROCOPT=G,	PROCOPT=G,
	KEYLEN=30	KEYLEN=30
	SENSEG	SENSEG
	NAME=S02SEG1,	NAME=S02SEG1,
	PARENT=0	PARENT=0
C -	PCB	PCB
*	TYPE=DB,	TYPE=DB,
	DBDNAME=DBD@M02A,	DBDNAME=DBD@M01A,
	PROCOPT=G,	PROCOPT=G,
	KEYLEN=30	KEYLEN=30
	SENSEG	SENSEG
	NAME=M02SEG1,	NAME=M02SEG1,
	PARENT=0	PARENT=0
D -	PCB	
	TYPE=DB,	
	DBDNAME=DBD@S03A,	
	PROCOPT=G,	
	KEYLEN=30	
D -	SENSEG	
	NAME=S03SEG1,	
	PARENT=0	
C -	PSBGEN	PSBGEN
	PSBNAME=PSB@003,	PSBNAME=PSB@003,
	LANG=PL/I,	LANG=PL/I,
	IOASIZE=600,	IOASIZE=600,
*	SSASIZE=840,	SSASIZE=280,
	IOEROPN=(100,WTOR),	IOEROPN=(100,WTOR),
	MAXQ=10,	MAXQ=10,
	OLIC=YES,	OLIC=YES,
	CMPAT=YES	CMPAT=YES
	END	END

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 201](#)
- [“Restrictions for the DBD/PSB/ACB Mapper utility” on page 202](#)
- [“Printing hierarchical structure of databases” on page 203](#)
- [“JCL requirements for the DBD/PSB/ACB Mapper utility” on page 203](#)
- [“Control statements for the DBD/PSB/ACB Mapper utility” on page 204](#)
- [“JCL examples for the DBD/PSB/ACB Mapper utility” on page 206](#)
- [“Output from the DBD/PSB/ACB Mapper utility” on page 209](#)

---

### DBD/PSB/ACB Mapper utility overview

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” on page 201](#)
- [“Program structure and job step” on page 202](#)
- [“Data flow” on page 202](#)

### 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 208](#)
- [“Example: Creating a PSB and generating a PSB map” on page 208](#)

## Program structure and job step

DBD/PSB/ACB Mapper consists of one program, FABMMAIN. This program controls load modules FABMDMAP, FABMPMAP, and FABMAMAP. The FABMMAIN 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.

FABMMAIN 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.

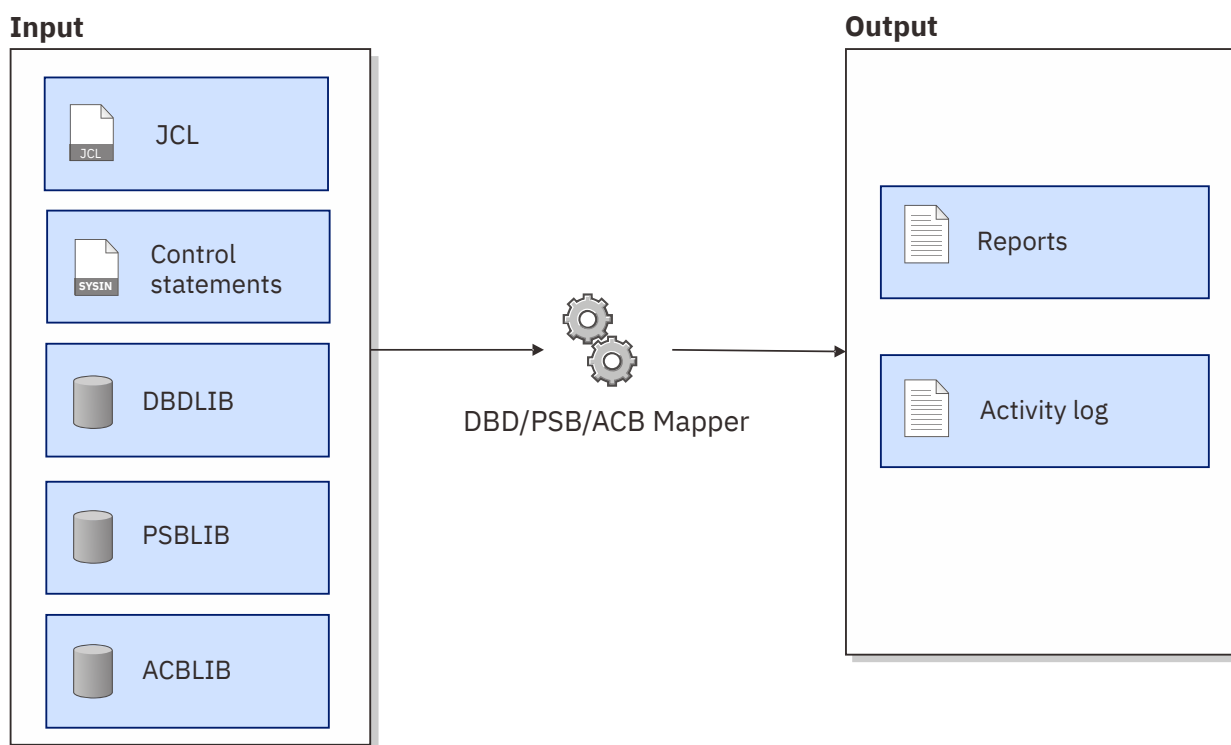


Figure 79. Data flow for DBD/PSB/ACB Mapper

## 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 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 203.
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 204.
3. Submit the job.
4. Check the output data sets that are generated.  
See [“Output from the DBD/PSB/ACB Mapper utility”](#) on page 209.

### Related reference

[JCL examples for the DBD/PSB/ACB Mapper utility](#)

This topic provides JCL examples for running the DBD/PSB/ACB Mapper utility to print maps of databases.

## 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”](#) on page 203
- [“EXEC statement”](#) on page 203
- [“DD statements”](#) on page 204

### JCL example

An example of the JCL that is required for DBD/PSB/ACB Mapper is shown in the following figure.

```
//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:

```
//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 204](#).

## 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” on page 204](#)
- [“Syntax rules” on page 205](#)
- [“Control statement keywords” on page 205](#)
- [“Quick reference for control statements and DD statements” on page 206](#)

### 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:

```
ACB=XXXXXXXX
PSB=XXXXXXXX
PSB=XXXXXXXX
ACB=XXXXXXXX
DBD=XXXXXXXX
ACBDBD=XXXXXXXX
```

## 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

Function	Control keyword	Required DD statements					
		SYSPRINT DD	SYSOUT DD	DBDLIB DD	PSBLIB DD	ACBLIB DD	SYSIN DD
DBD map	DBD=	Required	Required	Required			Required
PSB map	PSB=	Required	Required	Required	Required		Required
ACB map	ACB= ACBDBD=	Required	Required			Required	Required

## JCL examples for the DBD/PSB/ACB Mapper utility

This topic provides 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=FABMMAIN
//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=B000OUT01,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.



```

//PSBMAP EXEC PGM=FABMMAIN
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB,DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN 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=FABMMAIN
//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

```
//MAPPER EXEC PGM=FABMMAIN
//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 *
        DBD=BE1PARTS,X
        PSB=PE1CPINV,X
        PSB=PE1PPINV
        ACB=PE1CPINV
        PSB=PE1PARTS
        ACB=PE1CPPUR
        DBD=BE2PCUST
        ACB=PE1PARTS
        ACB=PE1PPINV
        DBD=B000INP01,X
        DBD=B000OUT01
/*
```

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 DSN=IMSVS.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

## 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),
// 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 DSN=IMSVS.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.PSBLIB(&MBR),DISP=SHR
//SYSUT1 DD UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),
// SPACE=(1024,(100,10),RLSE),DISP=(,DELETE)
//*
//PSBMAP EXEC PGM=FABMPMAP,PARM=(&MBR)
//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.

```

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB MAPPER          "MESSAGES"
5655-U08                                                     DATE: 10/01/2021  TIME: 17.46.49
FABM0001I CONTROL CARD SUPPLIED IS:  DBD=TESTDB1          PAGE :      1
FABM0001I CONTROL CARD SUPPLIED IS:  PSB=TESTPSB1        FABMMAIN - V2.R2
FABM0001I CONTROL CARD SUPPLIED IS:  ACB=TESTDB1
FABM0001I CONTROL CARD SUPPLIED IS:  ACB=TESTPSB1
FABM0003I TESTDB1  SELECTED, EXTENDED REPORT
FABM00034I MEMBER TESTDB1  PROCESSED
FABM0003I TESTPSB1 SELECTED, EXTENDED REPORT
FABM00034I MEMBER TESTPSB1 PROCESSED
FABM0009I TESTDB1  SELECTED, EXTENDED REPORT
FABM00034I MEMBER TESTDB1  PROCESSED
FABM0009I TESTPSB1 SELECTED, EXTENDED REPORT
FABM00034I MEMBER TESTPSB1 PROCESSED

```

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.

**Note:** Even when the IMS installed level is updated to 15.2 and ACB, DBD, and PSB are generated with IMS, IMS 15.1 is still used for such resources.

## 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” on page 210](#)
- [“Differences among DBD, PSB, and ACB maps” on page 211](#)
- [“Example of a DBD map \(single-page\)” on page 212](#)
- [“Example of a DBD map \(multiple-pages\)” on page 212](#)

## 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

Character	Explanation
*	The first data set group
+	The second data set group
"	The third data set group
.	The fourth data set group
=	The fifth data set group
-	The sixth data set group
#	The seventh data set group
%	The eighth data set group
;	The ninth data set group
'	The 10th data set group
V	Virtual logical child
C	Pointer and parent segment concatenation

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 9999 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 (PROCOPT 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.

```

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB MAPPER          "DBD MAP"
5655-U08                                                     DATE: 10/01/2021 TIME: 14.20.21
DBDNAME=PHDAM04 VOLUME=IMSVS DSN=IMSVS.DBDLIB             DBDGEN:08/10/2021 18.21
DBDMAP OF PHDAM04                                         ACCESS=PHDAM VSAM
                                                              PAGE: A
                                                              FABMDMAP - V2.R2
                                                              IMS V15.1
ALL SEGMENTS WITHIN A SINGLE DATA SET GROUP ARE DISPLAYED USING THE SAME CHARACTER FOR BOX DELIMITERS.
```

```

*****
* ROOTA *
*****001*
|-----|
|-----|
|-----|
+++++++ + " "
+ DEP1A + " DEP1B "
+++++++002+ " "003"
```

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.

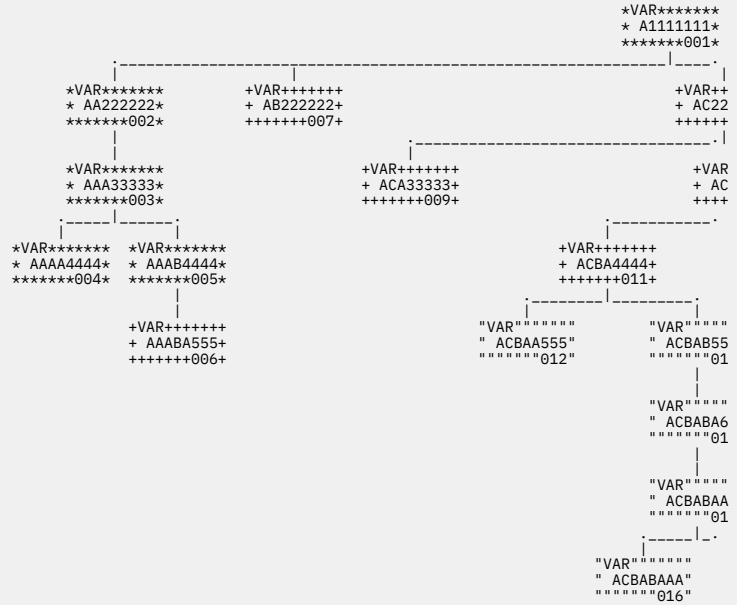


Figure 90. DBD map (multiple-pages) (Part 1 of 2)

ALL SEGMENTS WITHIN A SINGLE DATA SET GROUP ARE DISPLAYED USING THE SAME CHARACTER FOR BOX DELIMITERS.

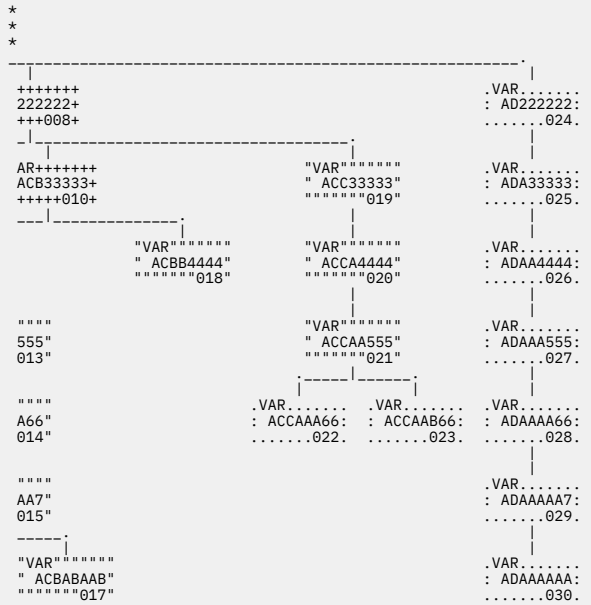


Figure 91. DBD map (multiple-pages) (Part 2 of 2)

## 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 214](#)
- [“Report field description” on page 215](#)

## Sample report

The following figures show an example of the PSB Summary report.

```

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB MAPPER          "PSB SUMMARY REPORT"
5655-U08                                                    DATE: 10/01/2021  TIME: 14.20.21
PSBNAME=PSBCRSIL  VOLUME=IMSVS  DSNAME=IMSVS.PSBLIB        PSBGEN:01/20/2021 11.12
PSB REPORT OF PSBCRSIL                                     PAGE: 1
                                                            FABMPMAP - V2.R2
                                                            IMS V15.1

PPPPPPPP          SSSS      BBBB      CCCC      RRRRRR      SSSS      IIIIIIIII  LLLL
PPPPPPPPPP        SSSSSSSS  BBBB      CCCCCCCC  RRRRRRRRR  SSSSSSSS  IIIIIIIII  LLLL
PP  PPP          SSS  SSS    BB  BBB    CCC  CCC    RR  RRR    SSS  SSS    II  LL
PP  PP          SS  SS     BB  BB    CC  CC     RR  RR    SS  SS     II  LL
PP  PP          SS  SS     BB  BB    CC  CC     RR  RR    SS  SS     II  LL
PP  PP          SS  SS     BB  BB    CC  CC     RR  RR    SS  SS     II  LL
PP  PPP          SSS      BBBB      CC          RR  RRR    SSS      II  LL
PPPPPPPP          SSSS      BBBB      CC          RRRRRRRR  SSSS      II  LL
PPPPPPPP          SSSS      BBBB      CC          RRRRRR    SSSS      II  LL
PP          SSS  SS     BB  BB    CC  CCC    RR  RRR    SSS      II  LL
PP          SS  SS     BB  BB    CC  CCC    RR  RRR    SS  SS     II  LL
PP          SS  SS     BB  BB    CC  CC     RR  RRR    SS  SS     II  LL
PP          SSS  SSS    BB  BBB    CCC  CCC    RR  RRR    SSS  SSS    II  LL
PPPP          SSSSSSSS  BBBB      CCCCCCCC  RRRR  RRRR  SSSSSSSS  IIIIIIIII  LLLLLLLLLL
PPPP          SSSS      BBBB      CCCC      RRRR  RRRR  SSSS      IIIIIIIII  LLLLLLLLLL

PSB PREFIX SUMMARY
      0 BYTES NEEDED FOR I/O WORK AREA
      0 BYTES NEEDED FOR SSA WORK AREA

ASSEMBLER/COBOL LANGUAGE OR NOT SPECIFIED

TP PCB SUMMARY
SYSTEM ADDS AN I/O PCB IN MPP AND BMP EXECUTION REGIONS.
NO USER TP PCB.

DB PCB SUMMARY
DB PCB #          1  DSCRSDVN

GSAM PCB SUMMARY
NO GSAM PCBs

TP PCB DETAIL
SYSTEM ADDS AN I/O PCB IN MPP AND BMP EXECUTION REGIONS.
NO USER TP PCB.

DB PCB DETAIL
DB PCB #          1  DSCRSDVN
PROCESSING OPTION = L
SINGLE POSITIONING REQUESTED
LENGTH OF KEY FEEDBACK AREA = 50      (000032 HEX)
NUMBER OF SENSITIVE SEGMENTS = 7

```

Figure 92. PSB Summary report (Part 1 of 2)



1	SSCRSP00	**ROOT**	L
2	SSCRSP12	SSCRSP00	L
3	SSCRSP21	SSCRSP12	L
4	SSCRSP22	SSCRSP12	L
5	SSCRSP23	SSCRSP12	L
6	SSCRSP13	SSCRSP00	L
7	SSCRSP14	SSCRSP00	L

GSAM PCB DETAIL  
NO GSAM PCBs

Figure 93. PSB Summary report (Part 2 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” on page 216](#)
- [“Example of the DBD report” on page 217](#)
- [“Example of the PSB report” on page 218](#)
- [“Report field description for the database information part” on page 219](#)
- [“Report field description for the data set group or DEDB area information part” on page 220](#)
- [“Report field description for the segment information part” on page 220](#)
- [“Report field description for the field information part” on page 222](#)

## 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 report is prepared. 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 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:

– \$SEGMnnn (nnn is a sequential number in a DBD)

– \$DBDnnn

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.

```

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB MAPPER          "DBD REPORT"
5655-U08                                                     DATE: 10/01/2021  TIME: 18.22.34          PAGE: 1
DBDNAME=DSCRSVDN  VOLUME=IMSVS  DSNAME=IMSVS.DBDLIB        DBDGEN:08/10/2021 18.21          FABMDMAP - V2.R2
DBDMAP OF DSCRSVDN                                         ACCESS=HDAM  VSAM                    IMS V15.1

          DDDD  SSS  CCC  RRRR  SSS  DDDD  V  V  N  N
          D  D  S  S  C  C  R  R  S  S  D  D  V  V  N  N
          D  D  S  C  R  R  S  D  D  V  V  NN  N
          D  D  SSS  C  RRRR  SSS  D  D  V  V  N  N  N
          D  D  S  C  R  R  S  S  D  D  V  V  N  NN
          D  D  S  S  C  C  R  R  S  S  D  D  V  V  N  N
          DDDD  SSS  CCC  R  R  SSS  DDDD  V  N  N

          VERSION=08/10/21 18.21
          LEVELS= 3          SEGMENTS= 8          DATA SET GROUPS= 2
          RANDOMIZING ROUTINE=DFSHDC40          ROOT ANCHOR POINTS=
ID= 1*  DSCRSVD0  PRIME DS LOG. RCD. LEN= 2041, BLOCKSIZE= 2048          SEGMENT LENGTH MAX= 80, MIN= 42
          OFLW DS LOG. RCD. LEN= 2041, BLOCKSIZE= 2048          KEY LENGTH MAX= 20, MIN= 20          NUMBSEG= 2
          VIRTSEG= 1

ID= 2+  DSCRSVD1  PRIME DS LOG. RCD. LEN= 2041, BLOCKSIZE= 2048          SEGMENT LENGTH MAX= 46, MIN= 20
          OFLW DS LOG. RCD. LEN= 2041, BLOCKSIZE= 2048          KEY LENGTH MAX= 20, MIN= 3          NUMBSEG= 5

SEG-NAME  SC#  LV  PAR  -LEN-  ---FREQ---  C  P  L  L  P  P  P  L  L  E  RULES  PHYS.  N-SEQ  SEG-NAME  D-B-NAME  FORM  LOG  CHLD
          R  FB  P  T  P  H  C  C  C  C  P  I  D  R  INSRT  OR  OR  OR  INSRT
          SSCRSP00*  1  1  0  50  0.00  X  X  FB  FB  .F  .L  .F  .L  S  I  D  R  LAST  *PFX*  LEN= 30  MAX= 50  PFX+MAX= 80
          *LC**  SSSCHP13  DSSCHHVN  SNGL  HERE  52
          *LC**  SSCRSP13  NONE  LAST
          *LC**  SSCRSP14  NONE  LAST
          *FLD*  ECOURSE  20  3  CHARACTER  SEQ, UNIQUE
          *FLD*  ECRSP0LE  2  1  CHARACTER

          SSCRSV11V  2  2  1  0  0.00
          *PR**  SSSCHP13  DSSCHHVN
          *FLD*  ESCH00LN  20  1  CHARACTER  SEQ, UNIQUE
          *SRC*  SSSCHP13  DSSCHHVN  DATA
          1 *PFX*  LEN= 26  MAX= 20  PFX+MAX= 46
          *LP**  SSCRSP00  DSCRSVDN  PHYS  NONE
          *FLD*  ECOURSEN  20  1  CHARACTER  SEQ, UNIQUE
          VAR  LEN
          *PFX*  LEN= 14  MAX= 20  PFX+MAX= 34
    
```

Figure 94. DBD report (Part 1 of 2)

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB MAPPER										"DBD REPORT"				PAGE: 2										
5655-U08										DATE: 10/01/2021 TIME: 18.22.34				FABMAP - V2.R2										
DBDNAME=DSCRSVDN VOLUME=IMSVS DSNAME=IMSVS.DBDLIB										DBDGEN:08/10/2021 18.21				IMS V15.1										
DBDMAP OF DSCRSVDN										ACCESS=HDAM VSAM														
SEG-NAME	SC#	LV	PAR	-LEN-	---	FREQ---	C	P	L	L	P	P	L	L	E	RULES	PHYS.	SEG-NAME	D-B-NAME	FORM	LOG	CHLD		
							T	T	P	T	P	H	C	C	C	C	P	OR	OR	OR	INSRT			
							R	FB	FB	FB	.	.	.	.	L	S	I	D	R	FLD-NAME	LEN	STRT	PNTR	RULES
SSCRSP21+	4	3	3	9		0.00	X	X	X							L V V	LAST	*LC** *FLD* *FLD*	SSFAC13 ECLASSID ECRSP12L	DSFACHON 6 2	NONE 3 1	LAST CHARACTER CHARACTER	22	
																		*PFX* *LP** *FLD* *FLD*	LEN= 14 SSSTUP00 EAPPLDAY ESTUDNUM	MAX= DSSTUIVN 6 3	9 PHYS NONE 4 1	PFX+MAX= CHARACTER CHARACTER FIX LEN	23 23	
SSCRSP22+	5	3	3	9		0.00	X	X	X							L V V	LAST	*PFX* *PR** *LP** *FLD* *FLD*	LEN= 14 SSFAC13 SSFAC00 EFROMDAT ETEACHNR	MAX= DSFACHON DSFACHON 6 3	9 VIRT NONE 4 1	PFX+MAX= CHARACTER CHARACTER FIX LEN	23 23	
SSCRSP23+	6	3	3	6		0.00	X	X	X							L V V	LAST	*PFX* *LP** *FLD* *FLD*	LEN= 14 SSCLSP00 ECLASSRO	MAX= DSCLSDVN 3	6 PHYS NONE 4 1	PFX+MAX= CHARACTER CHARACTER FIX LEN	20 20	
SSCRSP13*	7	2	1	20		0.00	XX	X	X	X						L V V	LAST	*PFX* *LP** *FLD*	LEN= 22 SSCRSP00 EREQUCRS	MAX= 20 20	20 PHYS NONE 1	PFX+MAX= CHARACTER CHARACTER FIX LEN	42 20	
SSCRSP14+	8	2	1	20		0.00		XX	X	XX						L V V	LAST	*PFX* *LP** *FLD*	LEN= 26 SSCRSP00 EREQBGRS	MAX= 20 20	20 PHYS NONE 1	PFX+MAX= CHARACTER CHARACTER FIX LEN	46 20	

Figure 95. DBD report (Part 2 of 2)

## Example of the PSB report

The following figure shows an example of the PSB report.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB MAPPER										"PSB REPORT"				PAGE: 1																		
5655-U08										DATE: 10/01/2021 TIME: 18.22.34				FABMAP - V2.R2																		
PSBNAME=PSBCRSIL VOLUME=IMSVS DSNAME=IMSVS.PSBLIB										PSBGEN:01/20/2021 11.12				IMS V15.1																		
DBDNAME=DSCRSVDN VOLUME=IMSVS DSNAME=IMSVS.DBDLIB										DBDGEN:08/10/2021 18.21				IMS V15.1																		
DBDMAP OF DSCRSVDN IN PSB-PSBCRSIL										ACCESS=HDAM VSAM																						
PPPP	SSS	BBBB	CCC	RRRR	SSS	IIIII	L											DDDD	SSS	CCC	RRRR	SSS	DDDD	V	V	N	N					
P	P	S	S	B	B	C	C	R	R	S	S	I	I	L				D	D	S	C	C	R	R	S	S	D	D	V	V	N	N
P	P	S	S	B	B	C	C	R	R	S	S	I	I	L	----			D	D	S	C	C	R	R	S	S	D	D	V	V	N	N
PPPP	SSS	BBBB	C	RRRR	SSS	I	L								----			D	D	SSS	C	RRRR	SSS	D	D	V	V	N	N			
P		S	S	B	B	C	C	R	R	S	S	I	I	L	----			D	D	S	C	C	R	R	S	S	D	D	V	V	N	N
P	S	S	S	B	B	C	C	R	R	S	S	I	I	L				D	D	S	C	C	R	R	S	S	D	D	V	V	N	N
P		SSS	BBBB	CCC	R	R	SSS	IIIII	LLLL									DDDD	SSS	CCC	R	R	SSS	DDDD	V	N	N					
LEVELS= 3										SEGMENTS= 7				DATA SET GROUPS= 2																		
RANDOMIZING ROUTINE=DFSHDC40										ROOT ANCHOR POINTS= 2				MAX RBN= 20																		
ID= 1*	DSCRSVD0	PRIME	DS	LOG.	RCD.	LEN=	2041,	BLOCKSIZE=	2048	SEGMENT LENGTH MAX= 80, MIN= 42																						
ID= 2*	DSCRSVD1	OFLW	DS	LOG.	RCD.	LEN=	2041,	BLOCKSIZE=	2048	KEY LENGTH MAX= 20, MIN= 20																						
		PRIME	DS	LOG.	RCD.	LEN=	2041,	BLOCKSIZE=	2048	SEGMENT LENGTH MAX= 46, MIN= 20																						
		OFLW	DS	LOG.	RCD.	LEN=	2041,	BLOCKSIZE=	2048	KEY LENGTH MAX= 20, MIN= 3																						
										NUMBSEG= 2																						
										NUMBSEG= 5																						
SEG-NAME	SC#	LV	PAR	-LEN-	---	FREQ---	C	P	L	L	P	P	L	L	E	RULES	PHYS.	SEG-NAME	D-B-NAME	FORM	LOG	CHLD										
							T	T	P	T	P	H	C	C	C	C	P	OR	OR	OR	INSRT											
							R	FB	FB	FB	.	.	.	.	L	S	I	D	R	FLD-NAME	LEN	STRT	PNTR	RULES								
SSCRSP00*	1	1	0	50		0.00	X	X								L V V	LAST	*PFX* *FLD* *FLD*	LEN= 30 ECOURSE ECRSP0LE	MAX= 20 2	50 3 1	PFX+MAX= CHARACTER CHARACTER	80 52									
SSCRSP12+	3	2	1	20		0.00	X	X	X							L V V	LAST	*PFX* *FLD* *FLD*	LEN= 14 ECLASSID ECRSP12L	MAX= 6 2	20 8 1	PFX+MAX= CHARACTER CHARACTER	34 22									
SSCRSP21+	4	3	3	9		0.00	X	X	X							L V V	LAST	*PFX* *FLD* *FLD*	LEN= 14 EAPPLDAY ESTUDNUM	MAX= 6 3	9 4 1	PFX+MAX= CHARACTER CHARACTER	23 23									
SSCRSP22+	5	3	3	9		0.00	X	X	X							L V V	LAST	*PFX* *FLD* *FLD*	LEN= 14 EFROMDAT ETEACHNR	MAX= 6 3	9 4 1	PFX+MAX= CHARACTER CHARACTER	23 23									
SSCRSP23+	6	3	3	6		0.00	X	X	X							L V V	LAST	*PFX* *FLD* *FLD*	LEN= 14 EFROMDAT ETEACHNR	MAX= 6 3	9 4 1	PFX+MAX= CHARACTER CHARACTER	23 23									

Figure 96. PSB report (Part 1 of 2)

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB MAPPER										"PSB REPORT"				PAGE: 2							
5655-U08										DATE: 10/01/2021				TIME: 18.22.34		FABMPMAP - V2.R2					
PSBNAME=PSBCRSIL VOLUME=IMSVS DSNAME=IMSVS.PSBLIB										PSBGEN:01/20/2021				11.12		IMS V15.1					
DBDNAME=DSCRSVFN VOLUME=IMSVS DSNAME=IMSVS.DBDLIB										DBDGEN:08/10/2021				18.21		IMS V15.1					
DBDMAP OF DSCRSVFN IN PSB-PSBCRSIL										ACCESS=HDAM				VSAM							
SEG-NAME	SC#	LV	PAR	-LEN-	---FREQ---	C T	P T	L P	L P	P C	L C	L C	L C	E P	RULES	PHYS. N-SEQ	SEG-NAME OR	D-B-NAME OR	FORM OR	LOG INSRT	CHLD RULES
SSCRSP13*	7	2	1	20	0.00	XX	X	X	X						L V V LAST	*FLD*	ECLASSRO	3	4	CHARACTER	SEQ,UNIQUE
																*PFX*	LEN= 22	MAX= 20	20	PFX+MAX=	42
																*FLD*	EREQUCRS	20	1	CHARACTER	SEQ,UNIQUE
SSCRSP14+	8	2	1	20	0.00		X	XX	X	XX					L V V LAST	*PFX*	LEN= 26	MAX= 20	20	PFX+MAX=	46
																*FLD*	EREQBCRS	20	1	CHARACTER	SEQ,UNIQUE

Figure 97. PSB report (Part 2 of 2)

## 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=NONE 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
    - FULLWORD
- 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:
  - \*XFD\* indexed field
  - Field name self-explanatory
  - SECONDARY INDEXED FIELD self-explanatory

- Following the XFD 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 225](#)
- [“Restrictions for the DBD/PSB/ACB Reversal utility” on page 227](#)
- [“Converting IMS control blocks to control statements” on page 228](#)
- [“JCL requirements for the DBD/PSB/ACB Reversal utility” on page 229](#)
- [“Control statements for the DBD/PSB/ACB Reversal utility” on page 231](#)
- [“JCL examples for the DBD/PSB/ACB Reversal utility” on page 241](#)
- [“Output from the DBD/PSB/ACB Reversal utility” on page 243](#)
- [“DBD/PSB/ACB Reversal Site Default Generation utility” on page 262](#)

---

### DBD/PSB/ACB Reversal utility overview

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” on page 225](#)
- [“Program structure” on page 226](#)
- [“Data flow” on page 226](#)

### 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 247.
- 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 229.

**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.

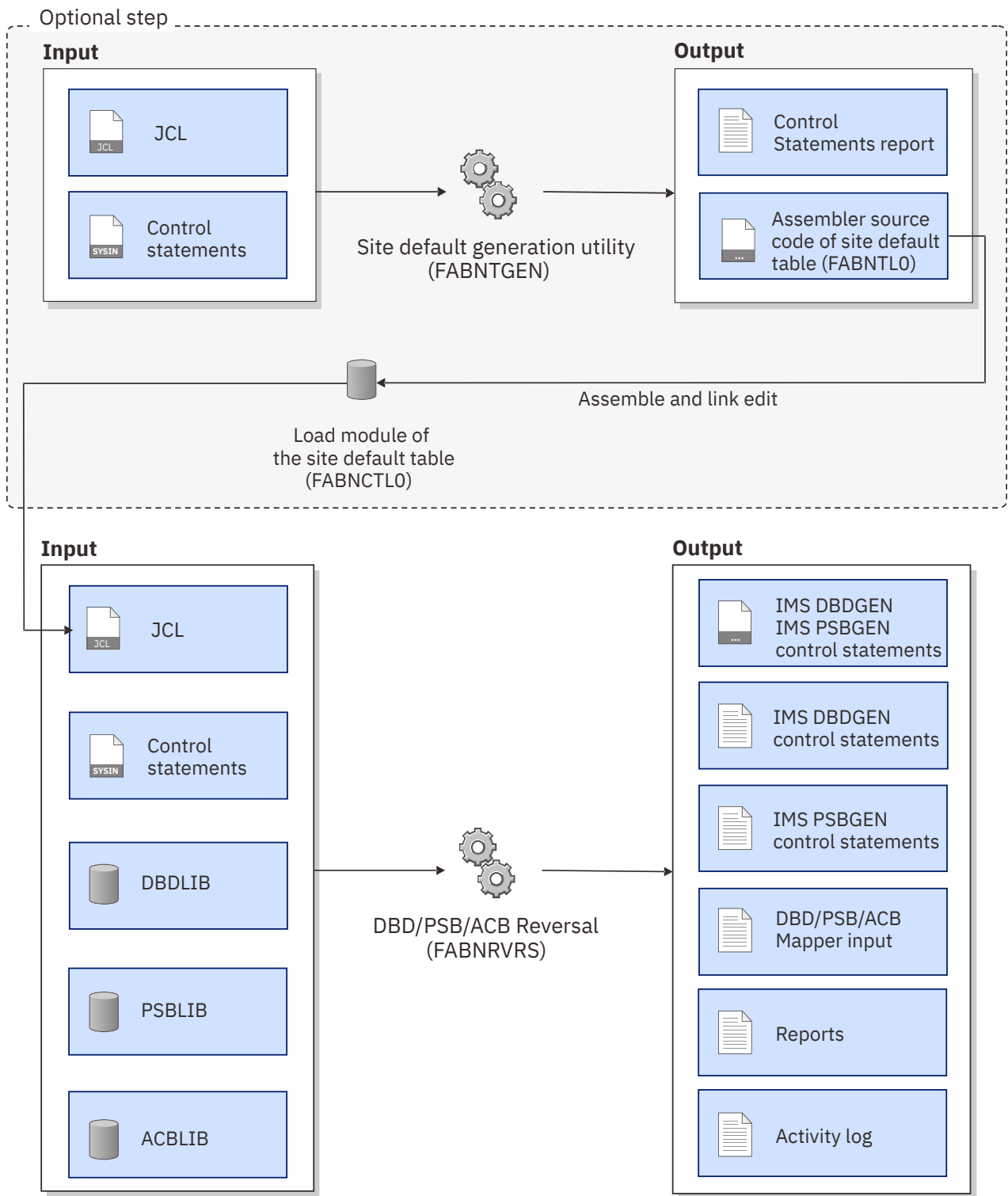


Figure 98. Data flow for DBD/PSB/ACB Reversal

## Restrictions for the DBD/PSB/ACB Reversal utility

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 245.

## 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 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 229.
- For a description about the Reversal Site Default Generation utility, see [“Reversal Site Default Generation utility overview”](#) on page 262.
- 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 262.

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 231.

3. Submit the job.

4. Check the output data sets that are generated.

See [“Output from the DBD/PSB/ACB Reversal utility”](#) on page 243.

### Related reference

[JCL examples for the DBD/PSB/ACB Reversal utility](#)

This topic provides JCL examples for running the DBD/PSB/ACB Reversal utility to re-create IMS DBDGEN and PSBGEN control statements.

## 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” on page 229](#)
- [“EXEC statement” on page 229](#)
- [“DD statements” on page 229](#)

### JCL example

An example of the JCL that is required for DBD/PSB/ACB Reversal is shown in the following figure.

```
//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:

```
//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 240](#).

#### **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 231](#).



## 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” on page 231](#)
- [“Syntax rules” on page 232](#)
- [“Runtime option control statements” on page 233](#)
- [“Function control statements” on page 236](#)
- [“Function keyword operands” on page 237](#)
- [“Function keyword options” on page 238](#)
- [“Quick reference for control statements and DD statements” on page 240](#)

## 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, SENSEG\_PROCOPT=YES, and REFER\_PSB=YES are applied to all subsequent function control statements that contain a DECODE function 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 ACBPSB ALL
  DECODE ACBPSB INCLUDE=member,member,...
  DECODE ACBPSB EXCLUDE=member,member,...
  DECODE ACBDBD ALL
  DECODE ACBDBD INCLUDE=member,member,...
  DECODE ACBDBD 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)

```
SEGREF DBD ALL SEGMENT=segname
SEGREF DBD INCLUDE=member,member,... SEGMENT=segname
SEGREF DBD EXCLUDE=member,member,... SEGMENT=segname
SEGREF PSB ALL SEGMENT=segname
SEGREF PSB INCLUDE=member,member,... SEGMENT=segname
SEGREF PSB EXCLUDE=member,member,... SEGMENT=segname
POPTREF PSB ALL SEARCHHDBD=member SEARCHOPT=procopt
POPTREF PSB INCLUDE=member,member,... SEARCHHDBD=member SEARCHOPT=procopt
POPTREF PSB EXCLUDE=member,member,... SEARCHHDBD=member SEARCHOPT=procopt
POPTREF ACB ALL SEARCHHDBD=member SEARCHOPT=procopt
POPTREF ACB INCLUDE=member,member,... SEARCHHDBD=member SEARCHOPT=procopt
POPTREF ACB EXCLUDE=member,member,... SEARCHHDBD=member SEARCHOPT=procopt
UNREF ACB
LISTLIB DBD
LISTLIB PSB
/*
```

Figure 101. Control statements for DBD/PSB/ACB Reversal (Part 2 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.

**FORMAT\_COL10=**

Specifies whether to print the decoded DBDGEN or PSBGEN macro statements starting at column 10. The utility prints one parameter per line, which starts at column 16. When the statement name is longer than 6 characters, one blank is placed between the DBDGEN or PSBGEN macro statements and the parameter that follows.

COMPRESS=YES and FORMAT\_COL10=YES cannot be enabled at once. When both of them are specified, the parameter specified last will take effect.

**YES**

The decoded DBDGEN or PSBGEN macro statements start at column 10.

**NO**

The decoded DBDGEN or PSBGEN macro statements are printed in the default format. This is the default value.

**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.

**REFER\_PSB=**

Specifies whether the utility skips the process to obtain the DBD VERSION or EXIT parameter value of an ACB for a DEDB or MSDB from a PSB-type ACB that references the ACB. When decoding an ACB for a DEDB or MSDB, a PSB-type ACB that references the ACB is used for obtaining the DBD VERSION or EXIT parameter value. When the target library has many members, it can be time-consuming to obtain these values. You can specify whether the utility skips this process to obtain these values. When the utility skips this process to obtain these parameter values, warning message FABN0077W is issued in both the decoded DBD source and the SYSOUT data set.

**YES**

The utility supplies DBD VERSION or EXIT parameter value of an ACB for a DEDB or MSDB from a PSB-type ACB that references the ACB. This is the default value.

**NO**

The utility does not supply these values from a PSB-type ACB.

**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 237](#).

### **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 POPTREF keyword. If this option is specified without the POPTREF 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 POPTREF keyword. The option must follow the SEARCHDBD option. If this option is specified without the POPTREF 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.

<b>PROCOPT criteria</b>	<b>Matched PROCOPT values</b>
SEARCHOPT=G	G

PROCOPT criteria	Matched PROCOPT values
SEARCHOPT=GO	GO
SEARCHOPT=AP	AP, PA
SEARCHOPT=*	Any PROCOPT value
SEARCHOPT=G*	G, GO, GP, GOP, GON, GHT
SEARCHOPT=P*	GP, GOP, AP, PA
SEARCHOPT=%	G, A
SEARCHOPT=G%	GO, GP
SEARCHOPT=P%	GP, PA, AP
SEARCHOPT=GO%	GOP, GON
SEARCHOPT=G%%	GOP, GON, GHT

## 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

Function	Control keyword	Control operand	Required and optional DD statements (O: Optional R: Required)											
			SYS PRINT	SYS OUT	SYS PUNCH	DBD SRC	PSB SRC	DBD LIB	PSB LIB	ACB LIB	MAP OUT	SYS IN	OPT PRT	
DBD reversal	DECODE	DBD		R	O <sup>1</sup>	O <sup>1</sup>		R				O <sup>4</sup>	R	O
PSB reversal	DECODE	PSB		R	O <sup>2</sup>		O <sup>2</sup>		R			O <sup>4</sup>	R	O
ACB reversal	DECODE	ACB		R	O	O <sup>3</sup>	O <sup>3</sup>	O <sup>5</sup>	O <sup>5</sup>	R		O <sup>4</sup>	R	O
ACB reversal (PSB-type only)	DECODE	ACBPSB		R	O <sup>2</sup>		O <sup>2</sup>		O <sup>5</sup>	R		O <sup>4</sup>	R	O
ACB reversal (DBD-type only)	DECODE	ACBDBD		R	O <sup>1</sup>	O <sup>1</sup>				R		O <sup>4</sup>	R	O
DBD summary	LIST	DBD	R	R				R					R	O
PSB summary	LIST	PSB	R	R					R				R	O
ACB summary	LIST	ACB	R	R				O <sup>5</sup>	O <sup>5</sup>	R			R	O
PCB PROCOPT	PROCOPT	PSB	R	R					R				R	O
PCB/ACB (PSB) PROCOPT	PROCOPT	ACB	R	R					O <sup>5</sup>	R			R	O
DBD-DBD XREF	XREF	DBD	R	R				R					R	O
PSB-DBD XREF	XREF	PSB	R	R					R				R	O
ACB(DBD)-ACB(DBD) XREF	XREF	ACB	R	R					O <sup>5</sup>	R			R	O
ACB(PSB)-ACB(DBD) XREF	XREF	ACB	R	R					O <sup>5</sup>	R			R	O
DDname XREF	DDNAMES	DBD	R	R				R					R	O
DDname XREF (ACB(DBD))	DDNAMES	ACB	R	R				O <sup>5</sup>	O <sup>5</sup>	R			R	O
DBD SEGMENT reference	SEGREF	DBD	R	R				R				O <sup>4</sup>	R	
PSB SEGMENT reference	SEGREF	PSB	R	R					R			O <sup>4</sup>	R	
PSB PROCOPT reference	POPTREF	PSB	R	R					R				R	O

Table 21. DBD/PSB/ACB Reversal functions, control statements, and DD statements (continued)

Function	Control keyword	Control operand	Required and optional DD statements (O: Optional R: Required)											
			SYS PRINT	SYS OUT	SYS PUNCH	DBD SRC	PSB SRC	DBD LIB	PSB LIB	ACB LIB	MAP OUT	SYS IN	OPT PRT	
ACB(PSB) PROCOPT reference	POPTREF	ACB	R	R						O	R		R	O
UN-REFERENCED ACB(DBD)	UNREF	ACB	R	R							R		R	
DBD library member list	LISTLIB	DBD	R	R				R					R	
PSB library member list	LISTLIB	PSB	R	R					R				R	

**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

This topic provides 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.SHPSLMD0,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
//SYSIN DD *
        DECODE ACB ALL
        DECODE ACBDBD 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

```
//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

```
//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.

```
//REVERS PROC MBR=TEMPNAME, RGN=2048K
//R      EXEC PGM=FABNRVRS, REGION=&RGN
//STEPLIB DD DSN=HPS.SHPSLMD0, DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB, DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB, DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSPUNCH DD SYSOUT=B
//MAPOUT DD UNIT=SYSDA, DISP=(,PASS),
//        SPACE=(80,(100,100),RLSE),
//        DCB=(BLKSIZE=80,RECFM=F,LRECL=80)
//SYSIN  DD DUMMY
//*
//M      EXEC PGM=FABMMAIN, REGION=&RGN
//STEPLIB DD DSN=HPS.SHPSLMD0, DISP=SHR
//DBDLIB DD DSN=IMSVS.DBDLIB, DISP=SHR
//PSBLIB DD DSN=IMSVS.PSBLIB, DISP=SHR
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN  DD DSN=* .R. MAPOUT, DISP=(OLD,DELETE)
//*
//      PEND
/*
```

Figure 106. Example of input generation functions of Reversal and Mapper

## 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 OPTPRT 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.

```

FABN0024I CONTROL CARD SUPPLIED IS : LIST DBD ALL
FABN0032I MEMBER TESTDB1 PROCESSED 18070000
FABN0024I CONTROL CARD SUPPLIED IS : LIST PSB ALL
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : LIST ACB ALL
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : PROCOPT PSB ALL
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : PROCOPT ACB ALL
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : XREF DBD ALL
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : XREF PSB ALL
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : XREF ACB ALL
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : DDNAMES DBD ALL
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : DDNAMES ACB ALL
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : SEGREF DBD ALL SEGMENT=SEGM1
FABN0026I MAPOUT FUNCTION SELECTED
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0052I SEGMENT NAME SEGM1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : SEGREF PSB ALL SEGMENT=SEGM1
FABN0026I MAPOUT FUNCTION SELECTED
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0052I SEGMENT NAME SEGM1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : POPTREF PSB ALL SEARCHDBD=TESTDB1 SEARCHOPT=G
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : POPTREF ACB ALL SEARCHDBD=TESTDB1 SEARCHOPT=G
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : DECODE DBD INCLUDE=TESTDB1
FABN0026I MAPOUT FUNCTION SELECTED
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : DECODE PSB INCLUDE=TESTPSB1
FABN0026I MAPOUT FUNCTION SELECTED
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : DECODE ACB INCLUDE=TESTPSB1
FABN0026I MAPOUT FUNCTION SELECTED
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0032I MEMBER TESTPSB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : DECODE ACBDBD INCLUDE=TESTDB1
FABN0026I MAPOUT FUNCTION SELECTED
FABN0032I MEMBER TESTDB1 PROCESSED
FABN0024I CONTROL CARD SUPPLIED IS : DECODE ACBPSB INCLUDE=TESTPSB1
FABN0026I MAPOUT FUNCTION SELECTED
FABN0032I MEMBER TESTPSB1 PROCESSED

```

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” on page 244](#)
- [“Restrictions on the generated control statements” on page 245](#)

### 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.

#### Notes:

- 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.
- Even when the IMS installed level is updated to 15.2 and ACB, DBD, and PSB are generated with IMS, IMS 15.1 is still used for such resources.

```

      TITLE 'ASSEMBLE OF DBDNAME=DSCLSDVN '
*      DSNAME=IMSVS.DBDLIB
*      VOL=IMSVS
*      DBDGEN DATE 11/22/2021 TIME 19.49
*      DECODE DATE 12/01/2021 TIME 10.18.54
*      IMS VERSION 15.1
      DBD      NAME=DSCLSDVN,ACCESS=(HDAM,VSAM),          C
              RMNAME=(DFSHDC40,1,2,6),PASSWD=YES,      C
              VERSION=          DATE 11/22/2021 TIME 19.49
*****
*      DATASET GROUP NUMBER 1
*****
DSG001 DATASET DD1=DSCLSDV0,SIZE=(2048),SCAN=3
*****
*      SEGMENT NUMBER 1
*****
      SEGM      NAME=SSCLSP00,PARENT=0,BYTES=(100,5),RULES=(LVV, LAST), C
              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, C
              RULES=HERE
*****
*      SEGMENT NUMBER 2
*****
      SEGM      NAME=SSCLSV11,PARENT=((SSCLSP00,)),PTR=PAIRED, C
              SOURCE=((SSCRSP23,DATA,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

```

      TITLE 'ASSEMBLE OF PSBNAME=PSBSMUAL '
*      DSNAME=IMSVS.PSBLIB
*      VOL=IMSVS
*      PSBGEN DATE 11/22/2021 TIME 19.50
*      DECODE DATE 12/01/2021 TIME 10.23.54
*      IMS VERSION 15.1
*****
*      PCB NUMBER 1      DB      NUMBER 1
*****
      PCB      TYPE=DB,DBDNAME=DSCLSDVN,PROCOPT=A,KEYLEN=29
      SENSEG   NAME=SSCLSP00,PARENT=0
      SENSEG   NAME=SSCLSV11,PARENT=SSCLSP00
*****
*      PCB NUMBER 1      DB      NUMBER 2
*****
      PCB      TYPE=DB,DBDNAME=DSSCHHVN,PROCOPT=A,KEYLEN=40
      SENSEG   NAME=SSSCHP00,PARENT=0
      SENSEG   NAME=SSSCHP11,PARENT=SSSCHP00
      SENSEG   NAME=SSSCHP12,PARENT=SSSCHP00
      SENSEG   NAME=SSSCHP13,PARENT=SSSCHP00
      PSBGEN   PSBNAME=PSBSMUAL,LANG=ASSEM,CMPAT=NO
      END

```

Figure 109. IMS PSBGEN 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 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 2.2, 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,LTERM=nnnnn, even if it is defined as PCB TYPE=TP,NAME=nnnnn, 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,\$DBDnnn)) (*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.

```

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL          "DBD XREF BY ACCESS - DBD NAME ORDER"          PAGE: 1
5655-U08                DATE: 10/01/2021  TIME: 10.57.49          FABNCODE - V2.R2

VOLUME=IMSVS    DSNAME=IMSVS.DBDLIB

  DBDNAME    ACCESS METHOD          DBDNAME    ACCESS METHOD          DBDNAME    ACCESS METHOD          DBDNAME    ACCESS METHOD
-----
HDAM01      HDAM,VSAM
HIDAM01     HIDAM,OSAM
INDEXDB     INDEX,VSAM,SGL
PHDAM01     PHDAM,OSAM
PHDAM04     PHDAM,VSAM
PHIDAM01    PHIDAM,OSAM
PHIDAM04    PHIDAM,VSAM
PSINDEX04   PSINDEX,VSAM
SINDEX01    INDEX,VSAM,SGL
  
```

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.

```

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL          "DBD XREF BY ACCESS - ACCESS ORDER"          PAGE: 1
5655-U08                DATE: 10/01/2021  TIME: 10.57.49          FABNCODE - V2.R2

VOLUME=IMSVS    DSNAME=IMSVS.DBDLIB

  ACCESS      COUNT    DBDNAME
-----
HIDAM,OSAM    1    HIDAM01
INDEX,VSAM,SGL 2    INDEXDB  SINDEX01
HDAM,VSAM     1    HDAM01
PHDAM,OSAM    1    PHDAM01
PHIDAM,OSAM   1    PHIDAM01
PHDAM,VSAM    1    PHDAM04
PHIDAM,VSAM   1    PHIDAM04
PSINDEX,VSAM  1    PSINDEX04
  
```

Figure 111. DBD XREF by Access - Access Order report

### XREF by Type reports for PSB and ACB(PSB)

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.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL 5655-U08			"PSB XREF BY TYPE - PSB NAME ORDER" DATE: 10/01/2021 TIME: 13.35.1			PAGE: 1 FABNCODE - V2.R2	
VOLUME=IMSVS DSNAME=IMSVS.PSBLIB							
PSBNAME	PSB TYPES	PSB LANGUAGE	PSBNAME	PSB TYPES	PSB LANGUAGE		
ALBBPG0	TP DB	ASSEMBLE/COBOL					
ALRFSEL0	TP DB	ASSEMBLE/COBOL					
CBSLOAD0	TP DB GSAM	ASSEMBLE/COBOL					
DBACHQ01	TP DB	ASSEMBLE/COBOL					
DBACHQ02	TP DB	ASSEMBLE/COBOL					
DBAMAP02	TP DB	ASSEMBLE/COBOL					
DBFSAMP2	DB	ASSEMBLE/COBOL					
DBTMP301	TP DB	ASSEMBLE/COBOL					
DBTMP302	TP DB	PL/I					
DBTMP303	DB	PL/I					
GSEDCTL0	TP DB GSAM	ASSEMBLE/COBOL					
KARENPSB	DB	ASSEMBLE/COBOL					
MPRTCUS0	TP DB	ASSEMBLE/COBOL					
NF2HA	TP DB	ASSEMBLE/COBOL					
NHGSAM01	GSAM	ASSEMBLE/COBOL					
NHGSAM02	GSAM	PL/I					
NHGSAM03	TP DB GSAM	PL/I					
PSBCLSil	DB	PL/I					
PSBCRSIL	DB	PL/I					
PSBP0010	TP DB GSAM	ASSEMBLE/COBOL					
PSBSMUAL	DB	PL/I					
PSBSMURE	DB	ASSEMBLE/COBOL					
PSBSMUUL	DB	ASSEMBLE/COBOL					
PSBSTUIL	DB	PL/I					
PSB01	TP DB GSAM	ASSEMBLE/COBOL					
PSB01A	TP	ASSEMBLE/COBOL					
PSB02	TP DB GSAM	ASSEMBLE/COBOL					
X080015T	TP DB	ASSEMBLE/COBOL					
ZDALK685	TP DB	ASSEMBLE/COBOL					

Figure 112. 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.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL 5655-U08			"PSB XREF BYTYPE - TYPE ORDER" DATE: 10/01/2021 TIME: 13.35.1			PAGE: 1 FABNCODE - V2.R2		
VOLUME=IMSVS DSNAME=IMSVS.PSBLIB								
PSB TYPES	COUNT	PSBNAME						
TP	1	PSB01A						
DB	9	DBFSAMP2 DBTMP303 KARENPSB PSBCLSil PSBCRSIL PSBSMUAL PSBSMURE PSBSMUUL PSBSTUIL						
TP DB	11	ALBBPG0 ALRFSEL0 DBACHQ01 DBACHQ02 DBAMAP02 DBTMP301 DBTMP302 MPRTCUS0 NF2HA X080015T						
		ZDALK685						
GSAM	2	NHGSAM01 NHGSAM02						
TP DB GSAM	6	CBSLOAD0 GSEDCTL0 NHGSAM03 PSBP0010 PSB01 PSB02						

Figure 113. 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.

```

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL          "DBD XREF BY DDNAME REPORT"          PAGE: 1
5655-U08                                                       DATE: 10/01/2021  TIME: 13.35.1      FABNDDN0 - V2.R2

VOLUME=IMSVS      DSNAME=IMSVS.DBDLIB

* REFERENCE REPORT *

DBDNAME           REFERENCES THE FOLLOWING DDNAMES
-----
ACSPA             ACSPA
BUDPA             BUDPA
CCPDI010          CCPDI610
CCPDP000          CCPDP500
CCP1I020          CCP1I620
CCP9P000          CCP9P500
CCRPA             CCRPA
CDGFGS01          CDGFGS01      CDGFGS01
CDIMAP00          CDIMAP00
CDPACT00          CDPACT00      CDPACT01      CDPACT02      CDPACT03
CDPAUD00          CDPAUD00
CDPCOL00          CDPCOL00
CDPDCS00          CDPDCS00
CDPHST00          CDPHST00
CDPMAP00          CDPMAP00      CDPMAP01
CDPTRN00          CDPTRN00
CDSMNM00          CDSMNM00
DBP1241           EP1241
DEB0001           DD01AR0
DEB0001           XPCW1D01
DEB0003           XPHS1D01
DSC LSDVN         DSC LSDV0      DSCRS0V1
DSCRS0V1          DSCRS0V0
DSFACH0N          DSFACH00
DSFACHVN          DSFACHV0
DSFDAXVN          DSFDAXV0      DSFDAXV1
DSNAMXVN          DSNAMXV0      DSNAMXV1
NHINDX01          DSNAMXV0      DSNAMXV1
DSSCHHVN          DSSCHHV0
DSSCHXIN          DSSCHXI0      DSSCHXI1
DSSTUIVN          DSSTUIV0      DSSTUIV1
EKTPA             EKTPA
GSAM03            GSAM03I0      GSAM03I0
HDAM2DSG          XPPR151E      XPPR152E
IVYPA             IVYPA
MNOPA             MNOPA
VALI1             VALI1
VALPA             VALPA

```

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.

VOLUME=IMSVS DSN=IMSVS.DBDLIB

\* REFERENCED REPORT \*

DDNAME	REFERENCED BY THE FOLLOWING DBDS
ACSPA	ACSPA
BUDPA	BUDPA
CCPDI610	CCPDI010
CCPDP500	CCPDP000
CCP1I620	CCP1I020
CCP9P500	CCP9P000
CCRPA	CCRPA
CDGFGS01	CDGFGS01
CDIMAP00	CDIMAP00
CDPACT00	CDPACT00
CDPACT01	CDPACT00
CDPACT02	CDPACT00
CDPACT03	CDPACT00
CDPAUD00	CDPAUD00
CDPCOL00	CDPCOL00
CDPDCS00	CDPDCS00
CDPHST00	CDPHST00
CDPMAP00	CDPMAP00
CDPMAP01	CDPMAP00
CDPTRN00	CDPTRN00
CDSMNM00	CDSMNM00
DD01AR0	DEBDD01
DSCLSDV0	DSCLSDVN
DSCRSDV0	DSCRSDVN
DSCRSDV1	DSCRSDVN
DSFACH00	DSFACH0N
DSFACXV0	DSFACXVN
DSFDAXV0	DSFDAXVN
DSFDAXV1	DSFDAXVN
DSNAMXV0	DSNAMXVN
DSNAMXV1	DSNAMXVN
DSSCHHV0	DSSCHHVN
DSSCHXI0	DSSCHXIN
DSSCHXI1	DSSCHXIN
DSSTUIV0	DSSTUIVN
DSSTUIV1	DSSTUIVN
EKTPA	EKTPA
EP1241	DBP1241
GSAM03I0	GSAM03
IVYPA	IVYPA
MNOPA	MNOPA
VALI1	VALI1
VALPA	VALPA
XPCW1D01	DEDB0001
XPHS1D01	DEDB0003
XPPR151E	HDAM2DSG
XPPR152E	HDAM2DSG

Figure 115. 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.

VOLUME=IMSVS DSN=IMSVS.PSBLIB

PSBNAME	PCB NO	TYPE	REF DBD / LTERM	PROCOPT	PROCSEQ(D)	PSBNAME	PCB NO	TYPE	REF DBD / LTERM	PROCOPT	PROCSEQ(D)
ALBBPG0	1	TP		N/A		DBAMAP02	13	DB	CDPMJR10	A	
	2	DB	CDPBMS00	GD			14	DB	CDPMAP00	GP	
ALRFSELO	1	TP		N/A			15	DB	CDPMRQ00	AP	
	2	DB	CDPRFP00	GOT			16	DB	CDSMTV00	G	
CBSLOAD0	1	TP		N/A			17	DB	CDPMTB00	GR	
	2	DB	CDPDCS00	I			18	DB	CDPMAP00	GRP	
	3	GSAM	CDGLOAD0	GS			19	DB	CDSMKR00	A	
	4	GSAM	CDGLOAD1	LS			20	DB	CDPMCS00	A	
DBACHQ01	1	TP		N/A			21	DB	CDSMCS00	A	
	2	DB	CDPCHQ00	AP			22	DB	CDSMCS10	A	
	3	DB	CDPCHQ10	AP			23	DB	CDSMCS20	A	
	4	DB	CDSCHQ00	G			24	DB	CDSMCS30	A	
	5	DB	CDSCHQ10	G			25	DB	CDPMCS00	GR	
	6	DB	CDSTRU00	G			26	DB	CDPMCS00	GR	
	7	DB	CDSTRU10	G			27	DB	CDPMCS00	GR	
	8	DB	CDSMCH00	G			28	DB	CDPMCS00	GR	
	9	DB	CDSMCH10	G		DBFSAMP2	1	DB	DBFSAMP3	A	
	10	DB	CDSDOC00	G		DBTMP301	1	TP	LTERMNAM	N/A	
	11	DB	CDSDOC10	G			2	DB	DEBDD01	GIRD	
	12	DB	CDSMCH00	G			3	DB	DEBDD02	A	
	13	DB	CDSMCH10	G		DBTMP302	1	TP		N/A	
	14	DB	CDSMCH00	G			2	DB	DEBDD01	AEP	
	15	DB	CDSMCH10	G		DBTMP303	1	DB	DBTDD303	A	
DBACHQ02	1	TP		N/A		GSEDC10	1	TP		N/A	
	2	DB	CDPCHQ00	GR			2	DB	CDPCCT00	G	
	3	DB	CDPCHQ10	GR			3	GSAM	CDGFGS01	GS	
	4	DB	CDPCHQ00	GR			4	GSAM	CDGFLS01	LS	
	5	DB	CDPCHQ10	GR			5	GSAM	CDGFLS02	LS	
	6	DB	CDPCHQ00	GR		KARENPSB	1	DB	EKTPA	G	
	7	DB	CDPCHQ10	GR			2	DB	GTLPA	G	
	8	DB	CDPCHQ00	GR			3	DB	ACSPA	G	
	9	DB	CDPCHQ10	GR			4	DB	ERRPA	G	
	10	DB	CDPCHQ00	GR			5	DB	BUDPA	G	
	11	DB	CDPCHQ10	GR			6	DB	IVMPA	G	
	12	DB	CDPCHQ00	GR			7	DB	ALNPA	G	
	13	DB	CDPCHQ10	GR			8	DB	CCRPA	G	
DBAMAP02	1	TP		N/A			9	DB	EMGPA	G	
	2	DB	CDPMAP00	AP			10	DB	IVYPA	G	
	3	DB	CDSMNM00	G			11	DB	LANPA	G	
	4	DB	CDSPAR00	G			12	DB	MNOPA	G	
	5	DB	CDSFIP00	G			13	DB	VALPA	G	
	6	DB	CDPMAP00	GRP		MPRTCUS0	1	TP		N/A	
	7	DB	CDPMAP00	GRP			2	DB	CDPMCS00	GOT	
	8	DB	CDPMAP00	GRP			3	DB	CDPMCS00	GOT	
	9	DB	CDPMTB00	AP			4	DB	CDPMCS00	GOT	
	10	DB	CDSMTB00	G			5	DB	CDPMCS00	GOT	
	11	DB	CDPMTB00	GRP			6	DB	CDPMCS00	GOT	
	12	DB	CDPMHP00	A			7	DB	CDPMAP00	GOTP	

Figure 116. PCB PROCOPT report

## 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.

### PROCOPT

This field shows the Processing Options (PROCOPT) 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.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL		"DBD TO DBD XREF REPORT"		PAGE: 1	
5655-U08		DATE: 10/01/2021 TIME: 13.35.1		FABNXREF - V2.R2	
VOLUME=IMSVS DSNAME=IMSVS.DBDLIB					
* REFERENCE REPORT *					
DBDNAME	REFERENCES	THE FOLLOWING DBDS			
BUDPA	IVMPA				
CCPDI010	CCPDP000				
CCPDL010	CCPDP000				
CCPDP000	CCPDI010	CCPDI020			
CCP1I020	CCP1P000				
CCP1L010	CCP1P000	CCP9P000			
CCP9P000	CCP9I010	CCP1P000			
CCRPA	IVYPA				
CDIMAP00	CDMPAP00				
CDLACT00	CDPACT00	CDPCHQ00	CDPCHQ10		
CDLMAP00	CDMPAP00				
CDPACT00	CDIACT00	CDSRSP00	CDPCHQ00	CDPCHQ10	CDSST00 CDSMOV00
CDPCOL00	CDSRAC00				
CDPDCS00	CDSOUT00	CDSPLT00	CDSL0S00		
CDPHST00	CDIHST00	CDSHST00	CDSHRP00		
CDMPAP00	CDIMAP00	CDSL0S00	CDSMNM00	CDSPAR00	CDSFIP00 CDSPST00 CDSMKR00 CDSSTA00 CDSADJ00
CDPTRN00	CDSTRA00	CDSTRT00			
CDSMNM00	CDMPAP00				
DSCLSDVN	DSCRSDVN				
DSCRSDVN	DSSCHHVN	DSFACHON	DSSTUIVN	DSCLSDVN	
DSCRSLBN	DSCRSDVN	DSSCHHVN	DSFACHON	DSSTUIVN	DSCLSDVN
DSFACHON	DSFACXVN	DSSCHHVN	DSCRSDVN	DSNMEXVN	DSFDAXVN
DSFACLBN	DSFACHON	DSSCHHVN	DSCRSDVN	DSSTUIVN	
DSFACXVN	DSFACHON				
DSFDAXVN	DSFACHON				
DSNAMXVN	DSSTUIVN	DSFACHON			
DSNMEXVN	DSSTUIVN	DSFACHON			
DSSCHHVN	DSSCHXIN	DSFACHON	DSSTUIVN	DSCRSDVN	
DSSCHLBN	DSSCHHVN	DSFACHON	DSCRSDVN	DSSTUIVN	
DSSCHXIN	DSSCHHVN				
DSSTUIVN	DSSCHHVN	DSCRSDVN	DSNAMXVN		
DSSTULBN	DSSTUIVN	DSSCHHVN	DSCRSDVN	DSFACHON	
IVYPA	IVMPA	MNOPA	CCRPA	AUNPA	
LGCL0001	HDAM0003	HDAM0002			
MNOPA	IVYPA	MNOI1	EMGPA	MNOI2	MNOI3
VALI1	VALPA				
VALPA	AUNPA	LANPA	VALI1		

Figure 117. DBD to DBD XREF - reference report

### Sample report: Referenced report

This report list 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.

VOLUME=IMSVS DSNAME=IMSVS.DBDLIB

\* REFERENCED REPORT \*

DBDNAME	REFERENCED BY THE FOLLOWING DBDS										
CCPDI010	CCPDP000										
CCPDP000	CCPDI010	CCPDL010									
CCP9P000	CCP1L010										
CCRPA	IVYPA										
CDIMAP00	CDPMAP00										
CDPACT00	CDLACT00										
CDPMAP00	CDIMAP00	CDLMAP00	CDSMNM00								
CDSMNM00	CDPMAP00										
DSC LSDVN	DSCRSDVN	DSCRSLBN									
DSCRSDVN	DSC LSDVN	DSCRSLBN	DSFACHON	DSFACLBN	DSSCHHVN	DSSCHLBN	DSSTUIVN	DSSTULBN	DSSCHHVN	DSSCHLBN	DSSTULBN
DSFACHON	DSCRSDVN	DSCRSLBN	DSFACHON	DSFACLBN	DSFACXVN	DSFDAXVN	DSNAMXVN	DSNMEXVN	DSSCHHVN	DSSCHLBN	DSSTULBN
DSFACXVN	DSFACHON										
DSFDAXVN	DSFACHON										
DSNAMXVN	DSSTUIVN										
DSNMEXVN	DSFACHON										
DSSCHHVN	DSCRSDVN	DSCRSLBN	DSFACHON	DSFACLBN	DSSCHLBN	DSSCHXIN	DSSTUIVN	DSSTULBN			
DSSCHXIN	DSSCHHVN										
DSSTUIVN	DSCRSDVN	DSCRSLBN	DSFACHON	DSFACLBN	DSNAMXVN	DSNMEXVN	DSSCHHVN	DSSCHLBN	DSSTULBN		
IVYPA	CCRPA	MNOPA									
MNOPA	IVYPA										
VALI1	VALPA										
VALPA	VALI1										

Figure 118. 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.



VOLUME=IMSVS DSN=IMSVS.PSBLIB

\* REFERENCE REPORT \*

PSBNAME	REFERENCES THE FOLLOWING DBDS									
ALBBPG0	CDPBMS00									
ALRFSELO	CDPRFP00									
CBSLOAD0	CDPDCS00	CDGLOAD0	CDGLOAD1							
DBACHQ01	CDPCHQ00	CDPCHQ10	CDSCHQ00	CDSCHQ10	CDSTRU00	CDSTRU10	CDESCSE00	CDESCSE10	CSDSOC00	CSDSOC10
	CDSMCH00	CDSMCH10	CDSRS00	CDSRS10						
DBACHQ02	CDPCHQ00	CDPCHQ10								
DBAMAP02	CDPMAP00	CDSMN00	CDSPAR00	CDSFIP00	CDPMTB00	CDSMTB00	CDPMHP00	CDPMJR10	CDPMRQ00	CDSMTV00
	CDSMKR00	CDPMCS00	CDSMCS00	CDSMCS10	CDSMCS20	CDSMCS30				
DBFSAMP2	DBFSAMD3									
DBTMP301	DEBDD01	DEBDD02								
DBTMP302	DEBDD01									
DBTMP303	DBTDD303									
GSEDTL0	CDPCCT00	CDGFGS01	CDGFLS01	CDGFLS02						
KARENPSB	EKTPA	GTLP	ACSPA	ERRPA	BUDPA	IVMPA	AUNPA	CCRPA	EMGPA	IVYPA
	LANPA	MNOPA	VALPA							
MPRTCUS0	CDPMCS00	CDPMAP00	CDPMRQ00							
NF2HA	CCPD1010	CCP11010								
NHGSAM01	GSAM01	GSAM02	GSAM03	GSAM04						
NHGSAM02	GSAM01	GSAM02	GSAM03	GSAM04						
NHGSAM03	DBP1241	DBP1242	DBP1243	GSAM01	GSAM02	GSAM03				
PSBCL5IL	DSCLSDVN									
PSBCRSIL	DSCRSDVN									
PSBP0010	CDPBNK00	CDGBP101								
PSBSMUAL	DSSCHHVN	DSFACHON	DSSTUIVN	DSCRSDVN	DSCLSDVN	DSNAMXVN	DSNMEXVN	DSFDAXVN	DSSCHLBN	DSFACLBN
	DSSTULBN	DSCRSLBN								
PSBSMURE	DSSCHHVN	DSFACHON	DSSTUIVN	DSCRSDVN	DSCLSDVN					
PSBSMUUL	DSSCHHVN	DSFACHON	DSSTUIVN	DSCRSDVN	DSCLSDVN					
PSBSTUUL	DSSTUIVN									
PSB01	DBP1241	GSAM02	GSAM03							
PSB02	DBP1241	GSAM02	GSAM03							
X080015T	HDAM0001	DEDB0001	DEDB0002	LGCL0001	HDAM0004	HDAM2DSG	DEDB0003	DEDB0004	DEDB14AR	
ZDALK685	CDSCHQ00	CDSTRU00	CDSCHQ10	CDSTRU10	CDPACT00	CDPCHQ00	CDPCHQ10			

Figure 119. 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.

VOLUME=IMSVS DSNAME=IMSVS.PSBLIB

\* REFERENCED REPORT \*

DBDNAME	REFERENCED BY THE FOLLOWING PSBS		
ACSPA	KARENPSB		
AUNPA	KARENPSB		
BUDPA	KARENPSB		
CCPDL010	NF2HA		
CCP1L010	NF2HA		
CCRPA	KARENPSB		
CDGPP101	PSBP0010		
CDGFGS01	GSEDCTL0		
CDGFLS01	GSEDCTL0		
CDGFLS02	GSEDCTL0		
CDGLOAD0	CBSLOAD0		
CDGLOAD1	CBSLOAD0		
CDPACT00	ZDALK685		
CDPBMS00	ALBBPG0		
CDPBK00	PSBP0010		
CDPCCT00	GSEDCTL0		
CDPCHQ00	DBACHQ01	DBACHQ02	ZDALK685
CDPCHQ10	DBACHQ01	DBACHQ02	ZDALK685
CDPDCS00	CBSLOAD0		
CDPMAP00	DBAMAP02	MPRTCUS0	
CDPMCS00	DBAMAP02	MPRTCUS0	
CDPMHP00	DBAMAP02		
CDPMJR10	DBAMAP02		
CDPMRQ00	DBAMAP02	MPRTCUS0	
CDPMTB00	DBAMAP02		
CDPRFP00	ALRFSEL0		
CDSCHQ00	DBACHQ01	ZDALK685	
CDSCHQ10	DBACHQ01	ZDALK685	
CDSCHR00	DBACHQ01		
CDSRS10	DBACHQ01		
CDSRS00	DBACHQ01		
CDSCE10	DBACHQ01		
CSDOC00	DBACHQ01		
CSDOC10	DBACHQ01		
CDSFIP00	DBAMAP02		
CDSMCH00	DBACHQ01		
CDSMCH10	DBACHQ01		
CDSMCS00	DBAMAP02		
CDSMCS10	DBAMAP02		
CDSMCS20	DBAMAP02		
CDSMCS30	DBAMAP02		
CDSMKR00	DBAMAP02		
CDSMNM00	DBAMAP02		
CDSMTB00	DBAMAP02		
CDSMTV00	DBAMAP02		
CDSPAR00	DBAMAP02		
CDSTRU00	DBACHQ01	ZDALK685	

Figure 120. PSB to DBD XREF - referenced report

## 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 256](#)
- [“Report field descriptions” on page 257](#)

## Sample report

The following figure shows an example of the DBD Segment Reference report.

VOLUME=IMSVS DSN=IMSVS.DBDLIB

SEGMENT NAME : \*

ORIGINAL: SEGMENT	DBD	REFERENCE: SEGMENT	DBD	ACCESS	STATEMENT	AS
PHIDAMD1	PHIDAM01	*	*	PHIDAM	SEGM NAME=	ORIGINAL
PHIDAMD2	PHIDAM01	* PHIDAMD3	* *	PHIDAM *	SEGM NAME= SEGM PARENT=	ORIGINAL PHYSICAL PARENT
PHIDAMD3	PHIDAM01	* PHIDAMD4	* *	PHIDAM *	SEGM NAME= SEGM PARENT=	ORIGINAL PHYSICAL PARENT
PHIDAMD4	PHIDAM01	*	*	PHIDAM	SEGM NAME=	ORIGINAL
PHIDAMRT	PHIDAM01	* PHIDAM1 PHIDAM2	* * *	PHIDAM * *	SEGM NAME= SEGM PARENT= SEGM PARENT=	ORIGINAL PHYSICAL PARENT PHYSICAL PARENT
SECDEP01	SECOND01	* SECDEP02 SECINDEX	* * SECINDEX	PHDAM * PSINDEX	SEGM NAME= SEGM PARENT= LCHILD NAME=	ORIGINAL PHYSICAL PARENT INDEX TARGET
SECDEP02	SECOND01	* SECDEP01	* *	PHDAM *	SEGM NAME= XDFLD SEGMENT=	ORIGINAL INDEX SOURCE
SECDEP03	SECOND01	*	*	PHDAM	SEGM NAME=	ORIGINAL
SECINDEX	SECINDEX	* SECDEP01	* SECOND01	PSINDEX PHDAM	SEGM NAME= LCHILD NAME=	ORIGINAL INDEX POINTER
SECROOT	SECOND01	* SECDEP01 SECDEP03	* * *	PHDAM * *	SEGM NAME= SEGM PARENT= SEGM PARENT=	ORIGINAL PHYSICAL PARENT PHYSICAL PARENT

Figure 121. DBD Segment Reference report

## 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 SEGM 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” on page 258](#)
- [“Report field descriptions” on page 258](#)

## Sample report

The following figure shows an example of the PSB Segment Reference report.

```
IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL          "PSB SEGMENT REFERENCE REPORT"
5655-U08                                                       DATE: 10/01/2021  TIME: 10.00.34
VOLUME=IMSVS  DSN=IMSVS.PSBLIB                                PAGE: 1
SEGMENT NAME : *                                             FABNSREF - V2.R2
```

SEGMENT	PSBNAME	PCB NO	REF DBD	PROCOPT	SEGMENT	PSBNAME	PCB NO	REF DBD	PROCOPT
HYS000A	HDAM@02L	1	HDAM@02	L					
SGIDE0A	HDAM@01L	1	HDAM@01	L					
	IDX@01G	1	HDAM@01	G					
XHYS000A	HDAM@01L	1	HDAM@01	L					
	IDX@01G	1	HDAM@01	G					
XSGIDE0A	HDAM@02L	1	HDAM@02	L					

Figure 122. PSB Segment Reference report

## Report field descriptions

The meaning of each column is as follows:

### SEGMENT

This field shows the segment name 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” on page 258](#)
- [“Report field descriptions” on page 259](#)

## Sample report

The following figure shows an example of the PSB PROCOPT Reference report.

PSBNAME	PCB NO	REF DBD	SEGMENT	PCB PROCOPT	SENSEG PROCOPT	PSBNAME	PCB NO	REF DBD	SEGMENT	PCB PROCOPT	SENSEG PROCOPT
DFHSAM05	1	DI21PART		G							
			PARTR00T		G						
			STANINFO		G						
			STOKSTAT		G						
			CYCCOUNT		G						
			BACKORDR		G						
DFSSAM02	1	DI21PART		G							
			PARTR00T		G						
			STANINFO		G						
DFSSAM03	1	DI21PART		G							
			PARTR00T		G						
			STANINFO		G						
			STOKSTAT		G						
			CYCCOUNT		G						
			BACKORDR		G						
DFSSAM07	1	DI21PART		G							
			PARTR00T		G						
			STANINFO		G						
			STOKSTAT		G						
			CYCCOUNT		G						
			BACKORDR		G						
DFSSAM08	1	DI21PART		G							
			PARTR00T		G						
			STANINFO		G						
			STOKSTAT		G						
			CYCCOUNT		G						
			BACKORDR		G						

Figure 123. PSB PROCOPT Reference report

## Report field descriptions

The meaning of each column is as follows:

### PSBNAME

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.

-----  
 CDGFGS01 CDIMAP00 CDPACT00 CDPAUD00 CDPDSC00 CDPHST00 CDPMAP00 CDPTRN00 CDSMNM00 DBP1241 DEDBDD01 DEDB0001 DEDB0003  
 DSCLSDVN DSCRSDVN DSFACHON DSFACXVN

Figure 124. Unreferenced ACB(DBD) report

## 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 260](#)
- [“Sample report: PSB library member list report” on page 260](#)
- [“Report field descriptions” on page 260](#)

### 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.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL				"DBD LIBRARY MEMBER LIST REPORT"		PAGE: 1	
5655-U08				DATE: 01/20/2021 TIME: 20.34.05		FABNDCOD - V2.R2	
DS#=001	VOLUME=IMSVS01	DSNAME=IMSVS.DBDLIB1					
DS#=002	VOLUME=IMSVS01	DSNAME=IMSVS.DBDLIB2					
MBRNAME	DS#	IMS LVL	SIZE (BYTES)	COMMENTS	DBDGEN DATE	TIME	
CDGFGS01	001	14.1	384		04/02/2020	15.58	
CDIMAP00	001	14.1	2,112		04/02/2020	15.58	
CDPACT00	001	14.1	416		04/02/2020	15.58	
CDPAUD00	001	14.1	1,712		04/02/2020	15.58	
DSSCHVHN	002	15.1	2,096		09/08/2021	15.34	
HDAM1DSG	002	15.1	1,880		09/08/2021	15.34	
HDAM2DS	002	15.1	400		09/08/2021	15.34	
HDAM2DSA	002	15.1	400	ALIAS	09/08/2021	15.34	

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.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL				"PSB LIBRARY MEMBER LIST REPORT"		PAGE: 1	
5655-U08				DATE: 01/20/2021 TIME: 20.40.11		FABNDCOD - V2.R2	
DS#=001	VOLUME=IMSVS01	DSNAME=IMSVS.PSBLIB1					
DS#=002	VOLUME=IMSVS01	DSNAME=IMSVS.PSBLIB2					
MBRNAME	DS#	IMS LVL	SIZE (BYTES)	COMMENTS	PSBGEN DATE	TIME	
PSBP000	002	15.1	328		04/02/2021	15.36	
PSBP010	002	15.1	1,328		04/02/2021	15.36	
PSBSMUUL	001	14.1	672		09/08/2020	15.58	
PSBSTUUL	001	14.1	272		09/08/2020	15.58	
PSBW01	002	14.1	152	NO DBDS REFERENCED	09/08/2020	15.58	
PSBW01A	002	14.1	152	ALIAS, NO DBDS REFERENCED	09/08/2020	15.58	

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. Even when the IMS installed level is updated to 15.2 and DBD and PSB are generated with IMS, IMS 15.1 is still used for such resources.

## SIZE (BYTES)

The size of the member record.

## COMMENTS

The following comments might be shown:

- ALIAS
- NO DBDS REFERENCED
- IMS 2.2 OR EARLIER

## MAPOUT data set

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)

## OPTPRT 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.

```
IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL          "RUN-TIME OPTION REPORT"          PAGE: 1
5655-U08                                                       DATE: 10/01/2021  TIME: 16.30.08          FABNDCOD - V2.R2
FUNCTION  MEMBER      TYPE  KEYWORD  OPTION
-----
DECODE   DBD@001A  DBD   DECOPT   COMPRESS=NO  COMMENT=NO
DECODE   DBD@002A  DBD   DECOPT   COMPRESS=NO  COMMENT=NO
DECODE   DBD@003A  DBD   DECOPT   COMPRESS=NO  COMMENT=NO
DECODE   PSB@001A  PSB   DECOPT   COMPRESS=NO  COMMENT=NO
DECODE   PSB@002A  PSB   DECOPT   COMPRESS=YES COMMENT=NO
DECODE   DBD@004A  DBD   DECOPT   COMPRESS=YES COMMENT=NO
DECODE   DBD@005A  DBD   DECOPT   COMPRESS=YES COMMENT=NO
DECODE   ACB@001A  ACB   DECOPT   COMPRESS=YES COMMENT=YES
DECODE   ACB@002A  ACB   DECOPT   COMPRESS=YES COMMENT=YES
DECODE   ACB@003A  ACB   DECOPT   COMPRESS=YES COMMENT=YES
```

Figure 128. Run-time Option report

## DBD/PSB/ACB Reversal Site Default Generation utility

---

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” on page 262](#)
- [“Reporting the SYSIN site default table” on page 262](#)

#### 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 FABNCTLO (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 (FABNCTLO).

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 262](#).

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 264](#).
- For a description of the control statements, see [“Control statements for the Reversal Site Default Generation utility” on page 266](#).

2. Assemble and link-edit the FABNCTLO source code.

To create the SYSIN site default table module FABNCTLO, 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 FABNCTLO 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 (FABNCTLO) 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 FABNCTLO.

```

//*****
//* FABNTGEN - DBD/PSB/ACB REVERSAL SITE DEFAULT GENERATION UTILITY
//* ( PARM='GEN' SAMPLE PROCEDURE )
//*****
//RVRSTGEN PROC HLQ='HPS'
//*-----
//* CREATE SOURCE CODE OF SITE DEFAULT TABLE
//*-----
//G EXEC PGM=FABNTGEN, PARM='GEN'
//STEPLIB DD DISP=SHR, DSN=&HLQ. . SHPSLMD0
//SYSPUNCH DD DISP=(NEW, PASS, DELETE), DSN=&&SOURCE,
// DCB=(RECFM=FB, BLKSIZE=800), SPACE=(TRK, (1, 1)), UNIT=SYSDA
//SYSOUT DD SYSOUT=*
//SYSUDUMP DD DUMMY
//*-----
//* ASSEMBLE & LINK ==> SITE DEFAULT TABLE MODULE (FABNCTLO)
//*-----
//ASM EXEC PGM=ASMA90, COND=(4, LT, G),
// PARM='OBJECT, NODECK, LIST, XREF(SHORT)'
//SYSLIN DD DISP=(, PASS), UNIT=SYSDA, SPACE=(CYL, (5, 5, 0)),
// DCB=(BLKSIZE=400), DSN=&&OBJECT
//SYSUT1 DD DISP=(, DELETE), UNIT=SYSDA, SPACE=(CYL, (10, 5))
//SYSPUNCH DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSIN DD DISP=(OLD, DELETE, DELETE), DSN=&&SOURCE
//*
//L EXEC PGM=IEWL, COND=(4, LT, ASM), REGION=4096K,
// PARM='LIST, REFR, REUS, AMODE=31, RMODE=ANY'
//SYSPRINT DD SYSOUT=*
//SYSLIN DD DISP=(OLD, DELETE, DELETE), DSN=&OBJECT
//*
// PEND
//*
//*-----*
//* FABNTGEN (PARM='GEN') - DBD/PSB/ACB REVERSAL *
//* SITE DEFAULT GENERATION UTILITY *
//*-----*
//GO EXEC RVRSTGEN, HLQ=HPS
//*-----*
//* SPECIFY SITE DEFAULT VALUES *
//*-----*
//G.SYSIN DD *
// DECOPT COMPRESS=NO, COMMENT=NO
//*
//L.SYSLMOD DD DISP=SHR, DSN=HPS.TABLELIB(FABNCTLO)
//*
//*-----*
//* FABNTGEN (PARM='REPORT') - DBD/PSB/ACB REVERSAL *
//* SITE DEFAULT GENERATION UTILITY *
//*-----*
//RVRSTGEN EXEC PGM=FABNTGEN, PARM='REPORT'
//STEPLIB DD DISP=SHR, DSN=HPS.TABLELIB
// DD DISP=SHR, DSN=HPS.SHPSLMD0
//SYSOUT DD SYSOUT=*
//SYSUDUMP DD DUMMY
//

```

Figure 129. Sample JCL for creating the site default table module FABNCTLO

## 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” on page 265](#)
- [“DD statement summary” on page 265](#)
- [“DD statements” on page 265](#)

## 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 22. FABNTGEN DD statements

DD name	Use	Format	EXEC PARM=	
			PARM='GEN'	PARM='REPORT'
STEPLIB	Input	PDS	Required	Required
SYSIN	Input	LRECL=80	Required	
SYSPUNCH	Output	LRECL=80	Required	
SYSOUT	Output	LRECL=133	Required	Required
SYSABEND or SYSUDUMP	Output	LRECL=133	Optional	Optional

## 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 (FABNCTLO).

### 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 23. Options for the site default in the SYSIN control statement*

<b>Keyword</b>	<b>Option (abbreviations)</b>	<b>Operand</b>
OPTION	ACB_GSAM	YES or NO
OPTION	ACB_REFERENCED	YES or NO
OPTION	PGM_COBOL	YES or NO
DECOPT	CHECK_LEN	YES or NO
DECOPT	COMMENT(C)	YES or NO
DECOPT	COMPRESS(COMP)	YES or NO
DECOPT	PCB_LABEL	YES or NO
DECOPT	SENSEG_PROCOPT	YES or NO
DECOPT	VERSION_GENDATE	YES or NO
DECOPT	FORMAT_COL10	YES or NO

For descriptions of the keyword and the options, see [“Control statements for the DBD/PSB/ACB Reversal utility”](#) on page 231.

## **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'.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL  
5655-U08

"MESSAGES"  
DATE: 10/01/2021 TIME: 14.35.21

PAGE: 1  
FABNTGEN - V2.R2

```
FABN1021I CONTROL CARD SUPPLIED IS: DECOPT COMMENT=NO
FABN1023I DECOPT OPTION USED: COMMENT=NO
FABN1021I CONTROL CARD SUPPLIED IS: DECOPT COMPRESS=NO
FABN1023I DECOPT OPTION USED: COMPRESS=NO
FABN1020I THE SOURCE CODE FOR THE SITE DEFAULT TABLE IS GENERATED
FABN1000I FABNTGEN ENDED NORMALLY
```

*Figure 130. Messages in the SYSOUT data set when PARM='GEN'*

The following messages are printed when the EXEC PARM parameter is 'REPORT'.

IMS LIBRARY INTEGRITY UTILITIES - DBD/PSB/ACB REVERSAL  
5655-U08

"MESSAGES"  
DATE: 10/01/2021 TIME: 14.35.21

PAGE: 1  
FABNTGEN - V2.R2

```
FABN1004I SITE DEFAULT OPTION USED: DECOPT COMMENT=YES
FABN1004I SITE DEFAULT OPTION USED: DECOPT COMPRESS=NO
FABN1030I SITE DEFAULT TABLE FABNCTLO IS PRINTED
FABN1000I FABNTGEN ENDED NORMALLY
```

*Figure 131. Messages in the SYSOUT data set when PARM='REPORT'*



---

## Chapter 9. MDA Reversal utility

The MDA Reversal utility converts DFSMDA members back into DFSMDA macros.

### Topics:

- [“MDA Reversal utility overview” on page 269](#)
- [“MDA Reversal utility restrictions” on page 270](#)
- [“Converting DFSMDA members back into DFSMDA macros” on page 271](#)
- [“JCL requirements for the MDA Reversal utility” on page 271](#)
- [“Control statements for the MDA Reversal utility” on page 272](#)
- [“JCL examples for the MDA Reversal utility” on page 275](#)
- [“Output from the MDA Reversal utility” on page 275](#)

---

### MDA Reversal utility overview

The MDA Reversal utility converts DFSMDA members back into DFSMDA macros. Also, the utility generates a report that lists DFSMDA members and their properties.

Subsections:

- [“Function overview” on page 269](#)
- [“Converting DFSMDA members back into DFSMDA macros” on page 269](#)
- [“Generating a report that contains a list of DFSMDA members” on page 269](#)
- [“Program structure” on page 269](#)
- [“Data flow” on page 270](#)

#### Function overview

The MDA Reversal utility provides the following functions:

- Converts DFSMDA members back into DFSMDA macros
- Generates a report that contains a list of DFSMDA members

#### Converting DFSMDA members back into DFSMDA macros

The utility converts DFSMDA members back into DFSMDA macros. This function reads one or more DFSMDA members from the specified library and converts them back into DFSMDA macros.

#### Generating a report that contains a list of DFSMDA members

The utility generates the Library Contents report, which contains a list of DFSMDA members found in the specified library. Detailed information about DFSMDA members, such as the name of the DD statement and the name of the data set, is shown for each DFSMDA member.

#### Program structure

The utility consists of one program, FABXMRVS, which controls other load modules and converts DFSMDA members back into DFSMDA macros. Based on the user specification, the program also generates reports about the specified library. This module uses a simple input format that is specified in the FABXMIN data set.

## Data flow

The following figure shows the general data flow for the MDA Reversal utility.

The input for the MDA Reversal utility is the FABXMIN data set, the MDALIB data set, and the ACBLIB data set for which sources are output and reports are created, and the output is DFSMDA macros, reports, and an activity log.

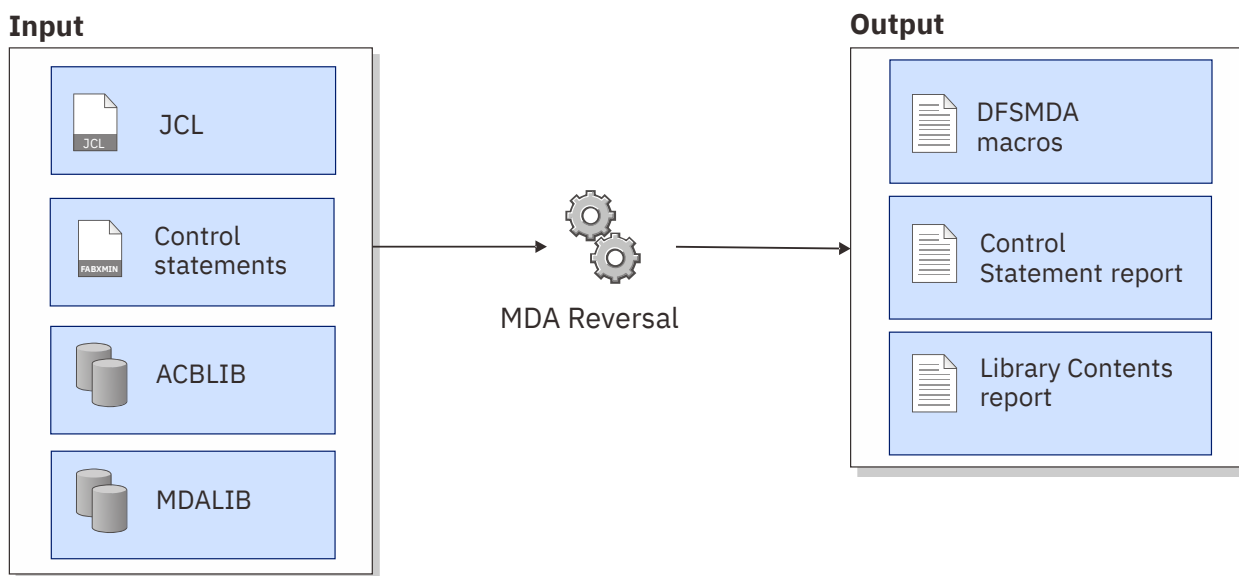


Figure 132. Data flow for the MDA Reversal utility

## MDA Reversal utility restrictions

Certain restrictions apply when you use the MDA Reversal utility.

If the following conditions are true, the TYPE parameters of the decoded DFSMDA macros differ from the original DFSMDA macro TYPE parameters. This is because DFSMDA members contain the same binary data for the following TYPE parameters and the MDA Reversal utility cannot distinguish the value that was originally set for the TYPE parameter.

- If the original DFSMDA macro was DFSMDA TYP=FPDEDB, the decoded DFSMDA macro will show DFSMDA TYPE=DATABASE.
- If the original DFSMDA macro was DFSMDA TYPE=RECON with an alternate DD name and WAIT=NO parameters, the decoded DFSMDA macros will show DFSMDA TYPE=DATABASE.

However, even if the value of the TYPE parameter is different, you can regenerate an identical DFSMDA member from the decoded DFSMDA macros.

If you want the original TYPE parameter values printed in decoded DFSMDA macros, specify the following options for the MDA Reversal utility control statements:

- Specify OPTION FPDEDB\_LIB to print DFSMDA TYPE=FPDEDB.
- Specify OPTION RECON\_ALT\_DD to print DFSMDA TYPE=RECON.



## Converting DFSMDA members back into DFSMDA macros

To convert DFSMDA members back into DFSMDA macros, you must prepare JCL for the MDA Reversal utility and submit the job.

### Before you begin

A sample JCL for the MDA Reversal utility is in the SHPSJCL0 library, member FABXMIVP. You can modify this sample JCL and then use it to run the utility.

### Procedure

1. In the MDA Reversal JCL, code the EXEC statement and DD statements.  
See [“JCL requirements for the MDA Reversal utility”](#) on page 271.
2. In the FABXMIN data set, code the control statements for the MDA Reversal utility.  
See [“Control statements for the MDA Reversal utility”](#) on page 272.
3. Submit the job.
4. Check the output data sets that are generated.  
See [“Output from the MDA Reversal utility”](#) on page 275.

### Related reference

[JCL examples for the MDA Reversal utility](#)

The figure in this topic shows a JCL example for converting DFSMDA members to DFSMDA macros with the MDA Reversal utility.

## JCL requirements for the MDA Reversal utility

When you code the JCL to run the MDA Reversal utility, include the EXEC statement and appropriate DD statements.

Subsections:

- [“JCL example”](#) on page 271
- [“EXEC statement”](#) on page 272
- [“DD statements”](#) on page 272

### JCL example

An example of the JCL that is required for MDA Reversal is shown in the following figure.

```
//JOB
//STEP      EXEC PGM=FABXMRVS
//STEPLIB  DD DISP=SHR,DSN=HPS.SHPSLMD0
//ACBLIB   DD DISP=SHR,DSN=PROD.ACBLIB
//DFSMDA   DD DISP=SHR,DSN=PROD.MDALIB
//FABXMSRC DD SYSOUT=B,FREE=CLOSE
//FABXMOUT DD SYSOUT=A
//MDASRC   DD DISP=SHR,DSN=PDS.MDASRC
//FABXMRPT DD SYSOUT=A
//FABXMIN  DD *
          PROC FUNC=DECODE
          OPTION MDA_LIST=YES,
                FPDEDB_LIB=ACBLIB,
                RECON_ALT_DD=RECON*
          MDA NAME=*
          END
/*
```

Figure 133. Example of MDA Reversal JCL

## EXEC statement

This statement must be in the following format:

```
//stepname EXEC PGM=FABXMRVS
```

## 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 IMS Library Integrity Utilities load module library data set.

### DFSMDA DD

Required input data set. Specify one or more libraries that contain the DFSMDA members for which DFSMDA macros and reports are created.

### ACBLIB DD

Required input data set when you specify the FPDEDB\_LIB=ACBLIB option for the OPTION statement to print the TYPE=FPDEDB parameter and DEDB database names.

### FABXMIN DD

Required input data set. Specify the data set that contains the control statements for the MDA Reversal utility.

The DCB parameters must be RECFM=FB, LRECL=80, and BLKSIZE must be a multiple of 80.

### FABXMOUT DD

Required output data set. Specify the data set for the Control Statement report.

The DCB parameters must be RECFM=FBA, LRECL=133, and BLKSIZE must be a multiple of 133.

### FABXMSRC DD

Optional output data set when you use the utility to decode DFSMDA members. Specify the data set for printing DFSMDA macros converted from DFSMDA members.

In this data set, the utility generates all DFSMDA macros converted from all the DFSMDA members that the utility identified from the specifications in the control statement.

The DCB parameters must be RECFM=FB, LRECL=80, and BLKSIZE must be a multiple of 80.

### MDASRC DD

Optional output data set when you use the utility to decode DFSMDA members. This data set must be a PDS or PDSE. The DFSMDA macros decoded in this data set are identical to those generated in the FABXMSRC data set, but in MDASRC data set, a data set member is created for each DFSMDA member.

The DCB parameters must be RECFM=FB, LRECL=80, and BLKSIZE must be a multiple of 80.

### FABXMRPT DD

Optional output data set. However, if you specify PROC FUNC=MDA\_LIST, which requests to create the Library Contents report without converting DFSMDA members to DFSMDA macros, this data set is a required data set. Specify the data set for the Library Contents report.

The DCB parameters must be RECFM=FBA, LRECL=133, and BLKSIZE must be a multiple of 133.

## Control statements for the MDA Reversal utility

---

The input for the MDA Reversal utility consists of control statements in the FABXMIN data set.

Subsections:

- [“Control statement example” on page 273](#)
- [“Syntax rules” on page 273](#)
- [“Statements and keywords” on page 273](#)

- [“Tips for using wildcard characters” on page 274](#)

## Control statement example

The following figure shows an example of the control statements for the MDA Reversal utility.

```
//FABXMIN DD *
PROC FUNC=DECODE
OPTION MDA_LIST=YES,
      FPDEDB_LIB=ACBLIB,
      RECON_ALT_DD=(RCN1,RCN2,
                  RCN3)
*
MDA NAME=HDAMDB1
MDA NAME=D*
END
/*
```

Figure 134. Control statement example for MDA Reversal

## 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.
- Option keywords 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.
- Option values in brackets must be separated by commas and must end with a closing parenthesis. If a comma is used instead of a closing parenthesis, the processing is continued to the next line.

## Statements and keywords

### PROC statement

Required statement. This statement must be coded on the first line with the following keyword and parameter:

#### **FUNC=DECODE**

Converts DFSMDA members back into DFSMDA macros.

#### **FUNC=MDA\_LIST**

Creates the Library Contents report without converting DFSMDA members to DFSMDA macros.

### OPTION statement

Optional statement. Specify this statement with one or more of the following keywords:

#### **FPDEDB\_LIB=ACBLIB**

This keyword is for DEDBs.

Prints FPDEDB for the DFSMDA TYPE parameter and the name of DEDB database in the output.

To specify this keyword, the following conditions must be met:

- The ACBLIB DD statement is specified in the JCL.
- The ACB library is consistent with the DFSMDA members.

When this keyword is specified, the utility obtains DEDB database names and DEDB area names from the ACB library. If the utility finds DEDB area names that match the names of DFSMDA members, it prints the following information:

- TYPE=FPDEDB in the DFSMDA macro statement.
- The name of the DEDB database (DBNAME=*dbname*) in the DFSMDA macro statement.
- The name of the DEDB database in the DBD column of the Library Contents report (if a Library Contents report is requested.)

**RECON\_ALT\_DD=ddname | (ddname1,ddname2,...)**

Specifies alternate DD names for the RECON data sets. This keyword is applicable only when the utility converts DFSMDA members to DFSMDA macros.

The utility identifies the DFSMDA member with the specified name and prints TYPE=RECON on the DFSMDA macro statement. You can specify up to nine DD names using brackets. You can also use wildcard characters to create a pattern-matching expression that specifies more than one DD names.

**MDA\_LIST=**

Specifies whether to generate the Library Contents report. This keyword is applicable only when the utility converts DFSMDA members to DFSMDA macros.

**YES**

The utility generates the Library Contents report. This is the default value.

**NO**

The utility does not generate the Library Contents report.

**MDA statement**

Required statement. Use this statement to select specific DFSMDA members. You can specify multiple MDA statements. Specify this statement with the following keyword:

**NAME=resource\_name**

Specify the name of a DFSMDA member. You can use wildcard characters to create a pattern-matching expression that specifies more than one DFSMDA members.

**END statement**

Optional statement. Use this statement to indicate the end of the control statements.

**Tips for using wildcard characters**

To specify multiple names, 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:

<b>Purpose</b>	<b>Coding example</b>
Select all DFSMDA members in the specified library	MDA NAME=*
Select DFSMDA members that have a name that begins with the letter D	MDA NAME=D*
Select DFSMDA members that begin with letters ABC, have any letter as the fourth character, and contain 001 as the fifth to seventh characters	MDA NAME=ABC%001

## JCL examples for the MDA Reversal utility

---

The figure in this topic shows a JCL example for converting DFSMDA members to DFSMDA macros with the MDA Reversal utility.

```
//JOB
//STEP      EXEC PGM=FABXMRVS
//STEPLIB  DD DISP=SHR,DSN=HPS.SHPSLMD0
//ACBLIB   DD DISP=SHR,DSN=PROD.ACBLIB
//DFSMDA   DD DISP=SHR,DSN=PROD.MDALIB
//FABXMSRC DD SYSOUT=B,FREE=CLOSE
//FABXMOUT DD SYSOUT=A
//MDASRC   DD DISP=SHR,DSN=PDS.MDASRC
//FABXMRPT DD SYSOUT=A
//FABXMIN  DD *
PROC FUNC=DECODE
OPTION MDA_LIST=YES,
      FPDEDB_LIB=ACBLIB,
      RECON_ALT_DD=(RCN1,RCN2,RCN3)
MDA NAME=HDAMDB1
MDA NAME=D*
END
/*
```

Figure 135. Example of converting DFSMDA members to DFSMDA macros

## Output from the MDA Reversal utility

---

Output from the MDA Reversal utility consists of the FABXMSRC data set, MDASRC data set, FABXMOUT data set, and the FABXMRPT data set.

### FABXMSRC data set

The FABXMSRC data set contains DFSMDA macros that are decoded from DFSMDA members.

### Decoded DFSMDA macros

The following figure shows an example of DFSMDA macros that the MDA Reversal utility generates.

```

*      DSNAMES=IMSVS.MDALIB1
*      DECODE DATE 10/01/2021 TIME 01.53.05
DFSM  TYPE=INITIAL
DFSM  TYPE=RECON,
      DDNAME=ALTRC1,
      WAIT=NO,
      DSNAMES=IMSVS.ALTRC1
DFSM  TYPE=RECON,
      DDNAME=ALTRC2,
      WAIT=YES,
      DSNAMES=IMSVS.ALTRC2
DFSM  TYPE=RECON,
      DDNAME=ALTRC3,
      WAIT=NO,
      DSNAMES=IMSVS.ALTRC3
DFSM  TYPE=CATDSHLQ,
      DDNAME=CAT1HLQ,
      SYSDSHLQ=IMSVS.CATDS11
DFSM  TYPE=DFSDCMON,
      DDNAME=IMSMON,
      DISP=SHR,
      UNIT=DASD,
      BUFNO=9,
      BLKSIZE=16384,
      DSNAMES=IMSVS.MONDS11
DFSM  TYPE=CATDBDEF,
      DBNAME=DFSHDBSC,
      DSNAMES=IMSVS.CATDB11
DFSM  TYPE=OLDS,
      DDNAME=DFSOLP00,
      DSNAMES=IMSVS.OLDSP00
DFSM  TYPE=OLDS,
      DDNAME=DFSOLS00,
      DSNAMES=IMSVS.OLDS00
DFSM  TYPE=TRACE,
      DDNAME=DFSTRA0T,
      UNIT=1234,
      BLKSIZE=16384,
      DSNAMES=IMSVS.TRCTP1
DFSM  TYPE=TRACE,
      DDNAME=DFSTRA01,
      DSNAMES=IMSVS.TRCD1
DFSM  TYPE=TRACE,
      DDNAME=DFSTRA02,
      DSNAMES=IMSVS.TRCD2
DFSM  TYPE=WADS,
      DDNAME=DFSWADS0,
      DSNAMES=IMSVS.WADSN00
DFSM  TYPE=WADS,
      DDNAME=DFSWADS1,
      DSNAMES=IMSVS.WADSN01
      DSNAMES=IMSVS.WADSN01

```

Figure 136. DFSMDA macros decoded by the MDA Reversal utility (Part 1 of 3)

```

DFSMDA TYPE=IMSACB, -
        DSNAME=IMSVS.IMSACB1
DFSMDA TYPE=IMSACBA -
DFSMDA TYPE=DATASET, -
        DDNAME=IMSACBA, -
        DISP=SHR, -
        DSNAME=IMSVS.IMSACBA1
DFSMDA TYPE=DATASET, -
        DDNAME=IMSACBA, -
        DISP=SHR, -
        DSNAME=IMSVS.IMSACBA2
DFSMDA TYPE=IMSACBB -
DFSMDA TYPE=DATASET, -
        DDNAME=IMSACBB, -
        DISP=SHR, -
        DSNAME=IMSVS.IMSACBB1
DFSMDA TYPE=SLDS, -
        DDNAME=IMSLOGR, -
        UNIT=DASD
DFSMDA TYPE=OLCSTAT, -
        DSNAME=IMSVS.OLCSTA1
DFSMDA TYPE=RECON, -
        DDNAME=RECON1, -
        WAIT=NO, -
        DSNAME=IMSVS.RECON1
DFSMDA TYPE=RECON, -
        DDNAME=RECON2, -
        WAIT=YES, -
        DSNAME=IMSVS.RECON2
DFSMDA TYPE=RECON, -
        DDNAME=RECON3, -
        WAIT=NO, -
        DSNAME=IMSVS.RECON3
DFSMDA TYPE=DATABASE, -
        DBNAME=TST@D01A
DFSMDA TYPE=DATASET, -
        DDNAME=TSTD01AA, -
        DISP=SHR, -
        DSNAME=IMSVS.TSTD01AA
DFSMDA TYPE=DATASET, -
        DDNAME=TSTD01AB, -
        DISP=OLD, -
        DSNAME=IMSVS.TSTD01AB
DFSMDA TYPE=DATASET, -
        DDNAME=TSTD01AC, -
        DISP=SHR, -
        DSNAME=IMSVS.TSTD01AC
DFSMDA TYPE=DATASET, -
        DDNAME=TSTD01AD, -
        DISP=OLD, -
        DSNAME=IMSVS.TSTD01AD

```

Figure 137. DFSMDA macros decoded by the MDA Reversal utility (Part 2 of 3)

```

DFSMDA TYPE=FPDEDB, -
        DBNAME=TST@E01A
DFSMDA TYPE=DATASET, -
DFSMDA TYPE=DATASET, -
        DDNAME=TSTE01AA, -
        DISP=SHR, -
        DSNAME=IMSVS.TSTE01AA
DFSMDA TYPE=FPDEDB, -
        DBNAME=TST@E01A
DFSMDA TYPE=DATASET, -
        DDNAME=TSTE01AB, -
        DISP=SHR, -
        DSNAME=IMSVS.TSTE01AB
DFSMDA TYPE=FPDEDB, -
        DBNAME=TST@E01A
DFSMDA TYPE=DATASET, -
        DDNAME=TSTE01AC, -
        DISP=SHR, -
        DSNAME=IMSVS.TSTE01AC
DFSMDA TYPE=FINAL -
END

```

Figure 138. DFSMDA macros decoded by the MDA Reversal utility (Part 3 of 3)





## FABXMRPT data set

The FABXMRPT data set contains the Library Contents report. This report lists information about the DFSMDA members found in the specified library.

Subsections:

- [“Sample report” on page 279](#)
- [“Report field descriptions” on page 279](#)

### Sample report

The following figure shows an example of the Library Contents report. DFSMDA DSN=*data\_set\_name* shows the library or libraries that contain the DFSMDA members listed in this report.

```
IMS LIBRARY INTEGRITY UTILITIES - MDA REVERSAL          "LIBRARY CONTENTS REPORT"          PAGE: 1
5655-U08                                               DATE: 10/01/2021  TIME: 01.53.05    FABXMRVS - V2.R2

DFSMDA DSN=IMSVS.MDALIB
-----
MDA MBR  DBD      DD/AREA  DISP DSNAME
-----
ALTRC1   ALTRC1   ALTRC1   SHR IMSVS.ALTRC1
ALTRC2   ALTRC2   ALTRC2   SHR IMSVS.ALTRC2
ALTRC3   ALTRC3   ALTRC3   SHR IMSVS.ALTRC3
CAT1HLQ  CAT1HLQ  CAT1HLQ  SHR IMSVS.CATDS11.BSDS
DFSDCMON DFSDCMON IMSMON   SHR IMSVS.MONDS11
DFSHBSC  DFSHBSC  DFSHBSC  SHR IMSVS.CATDB11
DFSOLP00 DFSOLP00 DFSOLP00 SHR IMSVS.OLDSP00
DFSOLS00 DFSOLS00 DFSOLS00 SHR IMSVS.OLDS00
DFSTRA0T DFSTRA0T DFSTRA0T SHR IMSVS.TRCTP1
DFSTRA01 DFSTRA01 DFSTRA01 SHR IMSVS.TRCDS1
DFSTRA02 DFSTRA02 DFSTRA02 SHR IMSVS.TRCDS2
DFSWADS0 DFSWADS0 DFSWADS0 SHR IMSVS.WADSN00
DFSWADS1 DFSWADS1 DFSWADS1 SHR IMSVS.WADSN01
IMSACB   IMSACB   IMSACB   SHR IMSVS.IMSACB1
IMSACBA  IMSACBA  IMSACBA  SHR IMSVS.IMSACBA1
IMSACBA  IMSACBA  IMSACBA  SHR IMSVS.IMSACBA2
IMSACBB  IMSACBB  IMSACBB  SHR IMSVS.IMSACBB1
IMSLOGR  IMSLOGR  IMSLOGR  SHR
OLCSTAT  OLCSTAT  OLCSTAT  SHR IMSVS.OLCSTA1
RECON1   RECON1   RECON1   SHR IMSVS.RECON1
RECON2   RECON2   RECON2   SHR IMSVS.RECON2
RECON3   RECON3   RECON3   SHR IMSVS.RECON3
TST@D01A TST@D01A TSTD01AA SHR IMSVS.TSTD01AA
          TST@D01A TSTD01AB OLD IMSVS.TSTD01AB
          TST@D01A TSTD01AC SHR IMSVS.TSTD01AC
          TST@D01A TSTD01AD OLD IMSVS.TSTD01AD
TSTE01AA TSTE01AA TSTE01AA SHR IMSVS.TSTE01AA
TSTE01AB TSTE01AA TSTE01AB SHR IMSVS.TSTE01AB
TSTE01AC TSTE01AA TSTE01AC SHR IMSVS.TSTE01AC
```

Figure 141. Library Contents report in the FABXMRPT data set

### Report field descriptions

The report contains the following fields:

#### MDA MBR

The DFSMDA member name.

#### DBD

This field shows one of the following resource names:

- If the DFSMDA member was generated with DFSMDA TYPE=DATABASE, this field shows the name of the database whose data sets are to be dynamically allocated.
- If the DFSMDA member was generated with DFSMDA TYPE=FPDEDB, this field shows the name of the DEDB area whose data sets are to be dynamically allocated.

If OPTION FPDEDB\_LIB is specified, this field shows the name of the DEDB database whose data sets are to be dynamically allocated.

- If the DFSMDA member was generated with other TYPE parameters, this field shows the name of the DFSMDA member.

#### DD/AREA

The name of the DD statement that defines the data set. This is the value specified by the DDNAME parameter of the DFSMDA macro.

**DISP**

Disposition of the allocated data set. This value was specified by the DISP parameter of the DFSMDA macro or was set as "SHR" when the DFSMDA member was generated.

**DSNAME**

The name of the data set specified by the DSNAME parameter of the DFSMDA macro. If the DFSMDA member was generated using a DFSMDA macro with the TYPE=CATDSDLQ parameter, this field shows the value specified by the SYSDSHLQ parameter with ".BSDS" added at the end.

---

## Chapter 10. Catalog Manager utility

The Catalog Manager utility helps you to analyze the definitions of IMS control blocks — databases (DBD) and application program views (PSBs) — in the IMS catalog and the IMS directory.

### Topics:

- [“Catalog Manager utility overview” on page 281](#)
- [“Catalog Manager utility restrictions” on page 284](#)
- [“Validating IMS control blocks in the IMS catalog” on page 286](#)
- [“Comparing IMS control blocks ” on page 286](#)
- [“Converting IMS control blocks to control statements” on page 287](#)
- [“Generating maps of IMS control blocks in the IMS catalog” on page 288](#)
- [“JCL requirements for the Catalog Manager utility” on page 288](#)
- [“Control statements for the Catalog Manager utility” on page 292](#)
- [“JCL examples for the Catalog Manager utility” on page 304](#)
- [“Output from the Catalog Manager utility” on page 311](#)

---

### Catalog Manager utility overview

The Catalog Manager utility analyzes IMS control blocks — databases (DBDs) and application program views (PSBs) — in the IMS catalog and in the IMS directory. The utility provides four functions; validate DBDs and PSBs, compare DBDs and PSBs, convert DBDs and PSBs to IMS DBDGEN and PSBGEN control statements, and generate DBD maps and PSB maps.

Subsections:

- [“Function overview” on page 281](#)
- [“Validating IMS control blocks in the IMS catalog and the IMS directory” on page 282](#)
- [“Comparing IMS control blocks” on page 283](#)
- [“Converting IMS control blocks to control statements” on page 283](#)
- [“Generating maps of IMS control blocks in the IMS catalog” on page 283](#)
- [“Program structure and job step” on page 283](#)
- [“Data flow” on page 283](#)

### Function overview

The Catalog Manager utility provides the following functions:

- Validating IMS control blocks in the IMS catalog and the IMS directory
- Comparing IMS control blocks
- Converting IMS control blocks to control statements
- Generating maps of IMS control blocks in the IMS catalog

These functions help you analyze IMS control blocks, specifically definitions of databases (DBDs) and program specification blocks (PSBs), in the IMS catalog and in the IMS directory.

IMS stores DBDs and PSBs in multiple locations; the IMS catalog, the IMS directory, ACB (application control block) libraries, DBD libraries, and PSB libraries. It is extremely important that DBDs and PSBs are consistent among these locations. In addition, when data sharing is enabled, DBDs and PSBs must be consistent across multiple IMS systems.

In order to ensure that DBDs and PSBs are consistent, you need to analyze DBDs and PSBs. However, DBDs and PSBs are not human-readable and their formats are different depending on where they are stored. In ACB libraries and the IMS directory, DBDs and PSBs are stored as binary format IMS control blocks, referred to as DBD-type ACBs and PSB-type ACBs. In the IMS catalog, DBDs and PSBs are stored as database records. This makes it more difficult to analyze DBDs and PSBs across multiple locations. The Catalog Manager utility analyzes DBDs and PSBs in different formats and helps you ensure that your DBDs and PSBs are maintained correctly.

Example use cases:

- If the IMS management of ACBs is enabled and you are maintaining DBDs and PSBs by populating the IMS catalog, you must always ensure that the IMS catalog and the IMS directory are in sync with ACB libraries. The validate function of the Catalog Manager utility helps you do so. If you find out that they are out of sync, you must run the ACB Generation and Catalog Populate utility (DFS3UACB) or the IMS Catalog Populate utility (DFS3PU00) to update the IMS catalog.
- If you are migrating from ACB libraries to the IMS management of ACBs, you can use the compare function of the Catalog Manager utility to verify that the definitions are correctly stored in the IMS catalog and the IMS directory. The compare function compares IMS control blocks in the IMS directory with those in the DBD, PSB, or ACB library, helping you ensure that the migration was done correctly.
- If data sharing is used, for instance, an IMS system (IMS-A) has the IMS management of ACBs enabled and the other IMS system (IMS-B) uses ACB libraries, you can use the compare function of the Catalog Manager utility to compare IMS control blocks between the IMS directory used by IMS-A and the ACB libraries used by IMS-B. If any differences are detected, you can identify the correct IMS control blocks from the compare reports, and also run the convert function to convert IMS control blocks to IMS DBDGEN or IMS PSBGEN control statements for further analysis. You can then run the DBDGEN, PSBGEN, or the ACBGEN utility and populate the IMS catalog by using the IMS Catalog Populate utility.

## Validating IMS control blocks in the IMS catalog and the IMS directory

The utility checks the ACB generation time stamps of DBDs and PSBs in the IMS catalog, the IMS directory, and ACB libraries to ensure that DBDs and PSBs are consistent across those resources. The utility generates several reports, including the IMS Catalog Validation report, which contains the results of the validation process. From this report, you can easily identify DBDs and PSBs that are inconsistent.

The utility also checks the consistency of the following information:

- If database versioning is enabled, whether the database version number is the same for each DBD.
- If the IMS management of ACBs is enabled, whether the time stamps of DBDs and PSBs in the IMS catalog, the IMS directory, and ACB libraries (if present) are consistent.
- If the IMS management of ACBs is not enabled, that is, when ACBs in the ACB libraries are used, whether the time stamps of the most recent instance of DBDs and PSBs in the IMS catalog and ACB libraries are the same.

When the IMS catalog is populated with the information from the ACB library, information about DBDs and PSBs in the ACB library is replicated in the IMS catalog. If the IMS management of ACBs is enabled, ACBs can be added to the IMS catalog and the IMS directory with the IMS Catalog Population utility (DFS3PU00). Regardless of how ACBs were added, information about DBDs and PSBs stored in the IMS catalog, the IMS directory (active and staging data sets), and ACB libraries (if used) must be kept in sync and consistent.

The IMS catalog can contain more than one instance for each DBD and PSB. The instances that the utility checks depend on whether the IMS management of ACBs is enabled or not:

- If the IMS management of ACBs is enabled, the utility checks whether the instance that has the same ACB generation time stamp as the ACB found in the IMS directory exists in the IMS catalog. Also, it checks whether the time stamp is consistent among the resources found in the IMS directory, IMS catalog, and the ACB library.
- If the IMS management of ACBs is not enabled, the utility checks whether the time stamp of the most recent instances found in the IMS catalog and ACB libraries are the same.

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 and the IMS directory are maintained correctly.

## Comparing IMS control blocks

The utility compares IMS control blocks — DBD-type ACBs (or DBDs) and PSB-type ACBs (or PSBs) — within the IMS directory, between the IMS directory and ACB libraries, and between the IMS directory and DBD libraries or PSB libraries. The utility generates several reports, including the Compare Listing, which reports differences (or similarities), the Compare Summary report, and the Error and Warning messages report.

The compare function is useful, for example, when you find inconsistencies in DBDs or PSBs with the validate function. You can use the compare function to identify the differences in IMS DBDGEN or IMS PSBGEN control statements by generating and reviewing the Compare Listing.

## Converting IMS control blocks to control statements

The utility converts IMS control blocks — DBD-type ACBs and PSB-type ACBs — in the IMS directory back into IMS DBDGEN or PSBGEN control statements. You can understand the definitions of ACBs in the IMS directory with the format of IMS DBDGEN or PSBGEN control statements.

## Generating maps of IMS control blocks in the IMS catalog

The utility produces and prints a pictorial layout, called a map, that graphically represents the structure and characteristics of a physical and logical IMS database. The Catalog Manager utility can also print a detailed report describing the characteristics of each DBD.

The utility provides the DBD map function which reads one or more DBDs from the IMS catalog database and produces maps and reports of the DBDs. The utility also provides the PSB map function which reads PSBs and DBDs that are referenced by PSBs in the IMS catalog database and produces maps and reports of the PSBs.

These maps and reports are similar to the maps and reports generated by the DBD/PSB/ACB Mapper utility.

## Program structure and job step

The Catalog Manager utility is provided as a z/OS 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 FABXCIN 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 FABXCRP0, FABXCRP1, FABXCRP2, and FABXCSRC DD statements. To learn more about which data sets are used by each function, see [“Output from the Catalog Manager utility”](#) on page 311.

## 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, DBDLIB, and PSBLIB data sets, the IMS catalog, and the IMS directory. The output consists of reports and messages.

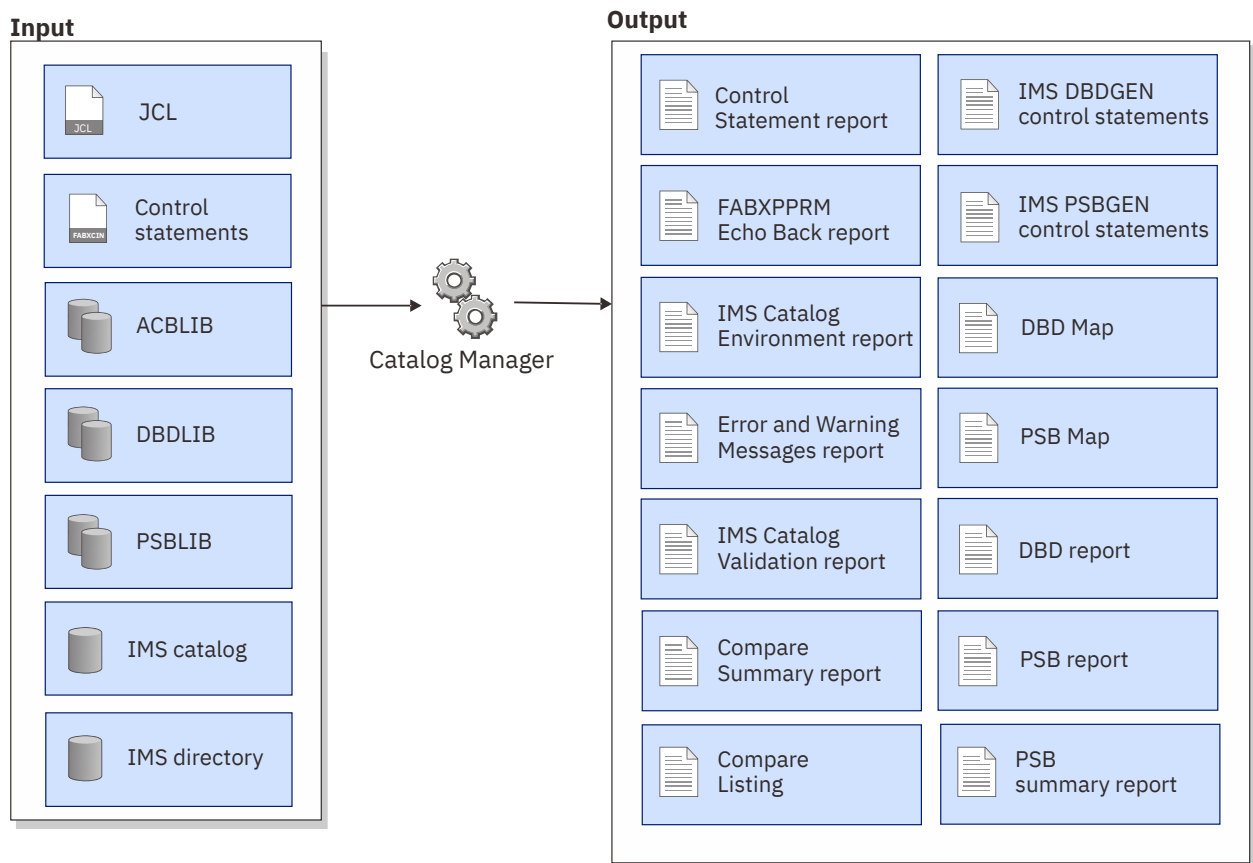


Figure 142. Data flow for the Catalog Manager utility

## Catalog Manager utility restrictions

Certain restrictions apply when you use the Catalog Manager utility.

The Catalog Manager utility has the following restrictions:

### Restrictions that apply to all the functions

IMS catalog was introduced with IMS 12. Therefore, the Catalog Manager utility supports IMS resources that are created by IMS 12 or later. However, the compare function supports DBDs and PSBs that were created by an earlier version of IMS and that are stored in DBD libraries or PSB libraries.

### Restrictions for the validate function

- If the IMS management of ACBs is not enabled or if DBDs and PSBs were generated by IMS 13, the Catalog Manager utility checks only the DBDs and PSBs that are found in the ACB libraries.
- DBDs and PSBs that are found only in the IMS catalog are not checked or included in the IMS Catalog Validation report.
- If you are using IMS 13, depending on the maintenance level of IMS, the Catalog Manager utility cannot process 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, apply APAR PI27237 to IMS 13.

## Restrictions for the compare function

- When the Catalog Manager utility compares DBD-type ACBs for DEDB, the utility also uses the PSB-type ACB that references the DBD to obtain DBD VERSION or EXIT parameter value. If a problem occurs when reading the PSB-type ACB, the DBD VERSION or EXIT parameter value is not compared.
- The Catalog Manager utility cannot process the following control blocks:
  - Control blocks for a logical database in the IMS directory because no control blocks for logical databases are stored in the IMS directory.
  - Control blocks for a GSAM or a logical database in the ACB library because no ACBs for GSAM or logical database are stored in the ACB library
  - Control blocks for a PSB that contains PCBs for a GSAM database in the ACB library because no PCBs for GSAM database are stored in the ACB library.
- The utility compares only the parameters that exist in the ACBs in ACB libraries. When the utility compares ACBs in the ACB library with those in the IMS directory, it ignores parameters that exist only in the ACBs in the IMS directory, such as GSAM PCBs.
- The reports generated by the compare function contain IMS DBDGEN and PSBGEN control statements. For the restrictions that apply to the generated control statements, see the following section.

## Restrictions for generated IMS DBDGEN and IMS PSBGEN control statements

The following restrictions apply to IMS DBDGEN control statements that the Catalog Manager utility generates:

- The order of the FIELD, LCHILD, and XDFLD statements that follow the SEGM statement is not the same as the user-required order in DBD control statements. The utility generates all 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, the utility converts the time stamp value to an Assembler comment statement.
- The utility cannot convert control blocks for a logical database in the IMS directory because no control blocks for a logical database are stored in the IMS directory.
- When the utility processes an ACB for DEDB, the utility also uses the PSB-type ACB that references the ACB to obtain the DBD VERSION or EXIT parameter value. If the utility processes an ACB in the IMS directory staging data set and the PSB-type ACB does not exist in the IMS staging data set, the DBD VERSION or EXIT parameter value is not converted. This restriction also applies to ACBs for MSDB but only for the DBD VERSION parameter value.
- The utility cannot print the FREQ parameter because ACBs in the IMS directory contain no information about the FREQ parameter of the SEGM statement.
- The utility cannot print the SIZE parameter, the second RECORD parameter, and the DEVICE parameter of the DATASET statement because ACBs in the IMS directory contain no information about these parameters.

The following restrictions apply to IMS PSBGEN control statements that the Catalog Manager utility generates:

- The PCB label and the PCBNAME parameter in the PCB statement are mutually exclusive. If the utility finds a PCB label parameter, the utility prints the value as a PCBNAME parameter. If you want the PCB label parameter printed in the output, specify the PCB\_LABEL=YES option for the FABXCIN control statement.
- If the PGM\_COBOL=YES option is not specified in the FABXCIN control statement, the utility prints the PSBGEN statement as PSBGEN LANG=ASSEM even if the statement is defined as PSBGEN LANG=COBOL or PSB LANG=, because there is no difference between the PSBs.
- The utility always prints the TP PCB statement as PCB TYPE=TP,LTERM=nnnn even if it is defined as PCB TYPE=TP,NAME=nnnn, because there is no difference between the PSBs.
- For a DEDB database, the utility always prints the POS parameter of the PCB statement as POS=S.

- The utility always prints the REPLACE parameter of the SENFLD statement as REPLACE=YES.
- The utility always prints the LIST parameter of the GSAM PCB statement as LIST=NO because GSAM PCBs in the IMS directory contains no information about the parameter.
- When the utility processes a PSB-type ACB which includes DBPCB for MSDB and the ACB for MSDB which is referenced by the PSB-type ACB does not exist in the processing IMS directory data set, the utility does not convert such PSB.

### Restrictions for the map function

The Catalog Manager utility can produce maps and reports of DBDs and PSBs from the IMS catalog database. It generates maps and reports with the values stored in the IMS catalog database. Some values of the generated report will be different from those of the DBD/PSB/ACB Mapper utility because some information is not stored in the IMS catalog database or the stored values are not the same. For details about differences between maps and reports, see [“FABXCRP2 data set \(Map function\)” on page 326](#).

## Validating IMS control blocks in the IMS catalog

---

To validate DBDs and PSBs in the IMS catalog by using the Catalog Manager utility, you must prepare JCL for the Catalog Manager utility and submit the job.

### Procedure

1. Create JCL for the Catalog Manager utility. You can copy sample JCL in the SHPSJCL0 library, member FABXCIVP, and modify it or create one of your own.
2. In the Catalog Manager utility JCL, code the EXEC statement and DD statements.  
See [“JCL requirements for the Catalog Manager utility” on page 288](#).
3. In the FABXCIN data set, code the control statements for the Catalog Manager utility.  
See [“Control statements for the Catalog Manager utility” on page 292](#).
4. Submit the job.
5. 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 validate function” on page 311](#).

### 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.

### Related reference

[Example: Validating DBDs and PSBs](#)

The figure in this topic shows a JCL example for validating DBDs and PSBs.

## Comparing IMS control blocks

---

The utility compares IMS control blocks — DBD-type ACBs (or DBDs) and PSB-type ACBs (or PSBs) — within the IMS directory, between the IMS directory and ACB libraries, and between the IMS directory and DBD libraries or PSB libraries.

### About this task

The compare function can compare active IMS control blocks in IMS directory data sets with staging IMS control blocks in a staging data set. It can also compare IMS directory with ACB, DBD, and PSB libraries. For example, you can use the compare function to:

- Identify which definitions will be changed by activating the staging IMS control block.



- Ensure that the IMS directory is in sync with ACB, DBD, PSB libraries. If differences are detected, you can correct them by running the IMS Catalog Populate utility (DFS3PU00).

To compare IMS control blocks, you must prepare JCL for the Catalog Manager utility and submit the job.

## Procedure

1. Create JCL for the Catalog Manager utility. You can copy the JCL example in [“Example: Comparing IMS control blocks”](#) on page 304 and modify it or create one of your own.
2. In the Catalog Manager utility JCL, code the EXEC statement and DD statements.  
See [“JCL requirements for the Catalog Manager utility”](#) on page 288.
3. In the FABXCIN data set, code the control statements for the Catalog Manager utility.  
See [“Control statements for the Catalog Manager utility”](#) on page 292.
4. Submit the job.
5. Check the job-step return code, WTO messages, and output data sets that are generated.

Refer to the Compare Summary report for a comparison summary. If any errors or warning messages were issued, see the Error and Warning messages report. These reports are generated in the FABXCRP1 data set.

To see the details of the differences detected, refer to the Compare Listing generated in the FABXCRP2 data set.

See [“Output from the compare function”](#) on page 317.

## Related reference

[Example: Comparing IMS control blocks](#)

The figures in this topic show JCL examples for the compare function.

## Converting IMS control blocks to control statements

---

The Catalog Manager utility can convert runtime ACBs in the IMS directory to IMS DBDGEN control statements or to IMS PSBGEN control statements. To convert IMS control blocks, you must prepare JCL for the Catalog Manager utility and submit the job.

## Procedure

1. Create JCL for the Catalog Manager utility. You can copy the JCL example in [“Example: Converting IMS control blocks to control statements”](#) on page 308 and modify it or create one of your own.
2. In the Catalog Manager utility JCL, code the EXEC statement and DD statements.  
See [“JCL requirements for the Catalog Manager utility”](#) on page 288.
3. In the FABXCIN data set, code the control statements for the Catalog Manager utility.  
See [“Control statements for the Catalog Manager utility”](#) on page 292.
4. Submit the job.
5. Check the job-step return code, WTO messages, and output data sets that are generated. The decoded source is generated in the FABXCSRC data set.  
See [“Output from the convert function”](#) on page 321.

## Related reference

[Example: Converting IMS control blocks to control statements](#)

The figure in this topic shows a JCL example for converting the runtime ACBs for databases and program views control blocks in the IMS directory to IMS DBDGEN and IMS PSBGEN control statements.

## Generating maps of IMS control blocks in the IMS catalog

---

The utility generates maps and reports of IMS control blocks - DBDs and PSBs - in the IMS catalog database.

### About this task

The map function can generate maps and reports of IMS control blocks in the IMS catalog database. These maps and reports are similar to the maps and reports of the DBD/PSB/ACB Mapper utility.

### Procedure

1. Create JCL for the Catalog Manager utility. You can copy the JCL example in [“Example: Generating maps and reports”](#) on page 308 and modify it or create one of your own.
2. In the Catalog Manager utility JCL, code the EXEC statement and DD statements.  
See [“JCL requirements for the Catalog Manager utility”](#) on page 288.
3. In the FABXCIN data set, code the control statements for the Catalog Manager utility.  
See [“Control statements for the Catalog Manager utility”](#) on page 292.
4. Submit the job.
5. Check the job-step return code, WTO messages, and output data sets that are generated.

Refer to the DBD map, PSB map, DBD report, PSB report, and PSB summary report. If any errors or warning messages were issued, see the Error and Warning messages report. These reports are generated in the FABXCRP1 data set.

To see some of the generated maps and reports, refer to these in the FABXCRP2 data set.

See [“Output from the map function”](#) on page 324.

### Related reference

[Example: Generating maps and reports](#)

The figures in this topic show JCL examples for the map function.

## JCL requirements for the Catalog Manager utility

---

When you code JCL for the Catalog Manager utility, include the EXEC statement and appropriate DD statements.

Subsections:

- [“JCL example”](#) on page 288
- [“EXEC statement”](#) on page 289
- [“DD statements”](#) on page 289

### JCL example

The following figure shows a JCL example that you can use to run the Catalog Manager utility program.

```

//CATMANJ JOB . . . .
//STEP EXEC PGM=FABXCATM
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMS15.SDFSRESL,DISP=SHR
//DFSRESLB DD DSN=IMS15.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
IOBF=(2048,4,N,N)
IOBF=(4096,4,N,N)
IOBF=(8192,4,N,N)
IOBF=(32000,4,N,N)
/*
//PROCLIB DD DSN=IMS15.PROCLIB
//RECON1 DD DSN=PROD.RECON1,DISP=SHR
//RECON2 DD DSN=PROD.RECON2,DISP=SHR
//*
//ACBLIB DD DSN=PROD.ACBLIB
//FABXPPRM DD *
DLI,FABXCATM,DFSCPL00,,,,,,,,,Y,N,,,,,,,,,DFSDF=CAT
//FABXCIN DD *
PROC FUNC=CHECK_GENTIME
DBD NAME=*
PSB NAME=*
END
/*

```

Figure 143. Catalog Manager utility example JCL: validating DBDs and PSBs

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. To use the following functions, you must also specify the SGLXLOAD library of IMS Tools Base 1.7 or later:

- Compare function

- Convert function
- Validate function (SGLXLOAD library is required only if you want to check the resources in IMS directory data sets)
- Map function (SGLXLOAD library is required when ACBMGMT=CATALOG is specified in the DFSDFxxx PROCLIB member)

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, IMS bootstrap data set, 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**

Optional input data set. If you specify the DLI keyword in the FABXPPRM data set, you must specify this DD statement. Specify the IMS.SDFSRESL data set.

#### **IMS DD**

Optional input data set. If you specify the DLI keyword in the FABXPPRM data set, you must specify this DD statement. Specify the PSB and DBD libraries that contain the DBDs and PSBs for the IMS catalog.

#### **DFSVSAMP DD**

Optional input data set. If you specify the DLI keyword in the FABXPPRM data set, you must specify this DD statement. 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 when using the utility to perform either of the following functions:

- Validate the consistency of DBDs and PSBs in the IMS catalog with ACBs in ACB libraries. Specify one or more ACB libraries that contain the DBD and PSB members to validate.
- Compare ACBs in the IMS directory with those in ACB libraries. Specify one or more ACB libraries that contain the DBD or PSB members to compare.

Optional input data set when using the utility to perform the following function:

- Generate maps of the IMS catalog database. Specify an active ACB library that contains DBD or PSB member to identify an active instance when the IMS catalog database does not have Active or Pending timestamp in its resource header segment and ACBMGMT=ACBLIB is specified in the DFSDFxxx PROCLIB member.

#### **ACBLIBS DD**

Optional input data set when using the utility to perform the following function:

- Generate maps of the IMS catalog database. Specify a staging ACB library that contains DBD or PSB member to identify a pending instance when the IMS catalog database does not have Active or Pending timestamp in its resource header segment and ACBMGMT=ACBLIB is specified in the DFSDFxxx PROCLIB member.

## DBDLIB DD

Required input data set when using the utility to compare ACBs in the IMS catalog with DBDs in DBD libraries. Specify one or more DBD libraries that contain the DBD members to compare with the IMS directory.

## PSBLIB DD

Required input data set when using the utility to compare ACBs in the IMS catalog with PSBs in PSB libraries. Specify one or more PSB libraries that contain the PSB members to compare with the IMS directory.

## FABXPPRM DD

Optional input statement. If you do not use the Catalog Definition exit routine (DFS3CDX0), you must specify this DD statement. Specify the parameters for the IMS region controller DFSRRC00.

Catalog Manager runs as a z/OS batch job, and it invokes DFSRRC00 to issue DL/I calls to the IMS catalog database. The parameters in FABXPPRM DD are given to DFSRRC00.

The format of the parameters 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*.

## DL/I

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,DFSCP000,,,,,,,,,Y,N,,,,,,,,,DFSDF=CAT  
/*
```

If you specify DLI and the IMS catalog is shared, you must specify IRLM support in the parameters. In the following example, the second Y and *irlmid* value indicate IRLM support:

```
//FABXPPRM DD *  
DLI,DFS3PU00,DFSCP000,,,,,,,,,Y,Y,irlmid,,,,,,,,,'DFSDF=001'  
/*
```

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,DFSCP000,,,,,,,,,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  
/*
```

## BMP

If the IMS control region is active on the same LPAR as the Catalog Manager utility job, you can specify the BMP keyword. Here is an example of the parameters:

```
//FABXPPRM DD *  
BMP,FABXCATM,DFSCP000,,,,,,,,,imsid,,,,,  
/*
```

When you specify the BMP keyword in the FABXPPRM data set, the Catalog Manager utility behaves as follows:

- Displays asterisks (\*\*\*) for some fields (such as IMS ID, ALIAS OF CATALOG DB) in the IMS Catalog Environment report.
- If the IMS control region is inactive, issues message DFS690A.

For both DL/I and BMP, you can specify any value for the second and third positional parameters because the Catalog Manager utility uses the following values for these parameters:

- The name of an internal LIU program for the second parameter.
- DFSCP000 for the third parameter.

#### **FABXCIN 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 292.](#)

#### **FABXCRP0 DD**

#### **FABXCRP1 DD**

#### **FABXCRP2 DD**

Optional output data sets. Specify these data sets for 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 311.](#)

#### **FABXSRC DD**

Optional output data set. This data set is used only for the convert function. Specify the data set in which the utility generates IMS DBDGEN control statements and IMS PSBGEN control statements. The DCB parameters must be RECFM=FB, LRECL=80, and BLKSIZE must be a multiple of 80.

#### **DBDSRC DD**

#### **PSBSRC DD**

Optional output data sets. These data sets are used only for the convert function. Specify the data sets in which the utility generates IMS DBDGEN control statements and IMS PSBGEN control statements. DBDSRC DD is for DBD members and PSBSRC DD is for PSB members. These data sets should be PDS or PDSE. The record format is fixed-blocked, and the logical record length is 80. The block size, if coded, must be a multiple of 80.

IMS DBDGEN and IMS PSBGEN control statements generated in these data sets are identical to those generated in the FABXSRC data set, but in DBDSRC and PSBSRC data sets, a data set member is created for each DBD or PSB.

#### **SYSUDUMP DD**

#### **SYSABEND DD**

#### **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.

Refer to the following topics for FABXCIN control statements:

- [“Control statements for the validate function” on page 293](#)
- [“Control statements for the compare function” on page 294](#)

- [“Control statements for the convert function” on page 298](#)
- [“Control statements for the map function” on page 301](#)

## Control statements for the validate function

Use the following information to prepare control statements for the validate function of the Catalog Manager utility.

Subsections:

- [“Syntax rules” on page 293](#)
- [“Control statement example” on page 293](#)
- [“Statements, keywords, and parameters” on page 293](#)
- [“Tips for using wildcard characters” on page 294](#)

### 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
```

### Control statement example

The following figure shows an example of the FABXCIN control statements to validate DBD and PSBs.

```
//FABXCIN DD *
PROC FUNC=CHECK_GENTIME
REPORT TIMESTAMP=FORMAT1
*
DBD NAME=HDAMDB1
PSB NAME=*
END
/*
```

Figure 144. Control statement example (validate function)

### Statements, keywords, and parameters

#### PROC statement

Required statement. This statement must be coded on the first line.

To invoke the validate function, specify: FUNC=CHECK\_GENTIME

The utility validates DBD and PSB members by comparing 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 or the IMS directory.

#### REPORT statement

Optional statement. Use this statement to specify the format of the time stamps printed in the Catalog Validation report.

#### TIMESTAMP=

Specify either of the following values:

#### FORMAT1

Time stamps are reported in the following format: *yyyy/mm/dd hh:mm:ss.th*. This is the default value.

## FORMAT2

Time stamps are reported in the following format: *yydddhmmssth*.

### DBD statement

Optional statement. Use this statement to select 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 select specific PSBs. Specify this statement with the following keywords and parameters:

#### **NAME=resource\_name**

Specify a PSB name. You can use wildcard characters to create a pattern-matching expression that specifies more than one PSB.

### END statement

Optional statement. Use this statement to indicate the end of the control statements.

## 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:

Purpose	Coding example
Validate all DBD members in the ACB libraries	DBD NAME=*
Validate DBDs that have a name that begins with the letter H	DBD NAME=H*
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	PSB NAME=ABC%001

## Control statements for the compare function

Use the following information to prepare control statements for the compare function of the Catalog Manager utility.

Subsections:

- [“Syntax rules” on page 294](#)
- [“Control statement example” on page 295](#)
- [“Statements, keywords, and parameters” on page 295](#)
- [“Tips for using wildcard characters” on page 298](#)

### 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*

## Control statement example

The following figure shows an example of the FABXCIN control statements to compare ACBs.

```
//FABXCIN DD *
PROC     FUNC=COMPARE,INPUT1=DIRECTORY_ACTIVE,INPUT2=ACBLIB
*
  DBD NAME1=HDAMDB1,NAME2=HDAMDB0
  DBD NAME1=TESTDB1
  PSB NAME1=PSB*
  END
/*
```

Figure 145. Control statement example (compare function)

## Statements, keywords, and parameters

### PROC statement

Required statement. This statement must be coded on the first line.

To invoke the compare function, specify: FUNC=COMPARE

The utility compares ACBs that are stored in the IMS directory. Depending on the values that you specify for the INPUTx keywords, the utility can compare active ACBs with active ACBs, active ACBs with staging ACBs, and staging ACBs with staging ACBs.

The utility can also compare ACBs in the IMS directory with ACBs in the ACB library, DBDs in the DBD library, or with PSBs in the PSB library.

FUNC=COMPARE must be accompanied with the following keywords and parameters. The utility compares resources in the library specified by the INPUT1 keyword with the library specified by the INPUT2 keyword.

**Note:** Abbreviations are shown in parentheses.

### INPUT1=

Specify either of the following values to indicate whether to use active ACBs or staging ACBs in the IMS directory.

#### **DIRECTORY\_ACTIVE (DIR\_ACT)**

Specifies to compare active ACBs in the IMS directory data sets.

#### **DIRECTORY\_STAGING (DIR\_STG)**

Specifies to compare staging ACBs in the staging data set.

### INPUT2=

Specify one of the following values:

#### **DIRECTORY\_ACTIVE (DIR\_ACT)**

Specifies to compare active ACBs in the IMS directory data sets.

#### **DIRECTORY\_STAGING (DIR\_STG)**

Specifies to compare staging ACBs in the staging data set.

#### **ACBLIB**

Specifies to compare ACBs in the ACB library.

#### **DBDLIB**

Specifies to compare DBDs in the DBD library.

#### **PSBLIB**

Specifies to compare PSBs in the PSB library.

### DBD statement

Optional statement. Use this statement to select specific DBDs. Specify this statement with the following keywords and parameters:

**NAME1=resource\_name**

**NAME2=resource\_name**

Specify a DBD name.

NAME1 specifies the resource name for the library that the INPUT1 statement specifies. NAME2 specifies the resource name for the library that the INPUT2 statement specifies. If NAME2 is omitted, the value you specify for NAME1 is used for NAME2.

For example, the following control statements are for comparing DBD DBDHDAM in the IMS directory active data sets with DBD DBDHDM2 in the ACB library.

```
PROC   FUNC=COMPARE, INPUT1=DIRECTORY_ACTIVE, INPUT2=ACBLIB
DBD    NAME1=DBDHDAM, NAME2=DBDHDM2
```

If you specify DBDLIB for the INPUT2 keyword, the resource names you specify for NAME1 and NAME2 keywords must be the same.

For both NAME1 and NAME2, you can use wildcard characters to create a pattern-matching expression that specifies more than one DBD. The following restrictions apply to using wildcard characters:

- For NAME1, you can use wildcard characters only when NAME2 keyword is omitted.
- For NAME2, you can use wildcard characters only when wildcard characters are not used for the NAME1 keyword value.

### PSB statement

Optional statement. Use this statement to select specific PSBs. Specify this statement with the following keywords and parameters:

**NAME1=resource\_name**

**NAME2=resource\_name**

Specify a PSB name.

NAME1 specifies the resource name for the library that the INPUT1 statement specifies. NAME2 specifies the resource name for the library that the INPUT2 statement specifies. If NAME2 is omitted, the value you specify for NAME1 is used for NAME2.

For example, the following control statements are for comparing PSB PSB001 in the IMS directory active data set with PSB PSB002 in the ACB library.

```
PROC   FUNC=COMPARE, INPUT1=DIRECTORY_STAGING, INPUT2=ACBLIB
PSB    NAME1=PSB001, NAME2=PSB002
```

If you specify PSBLIB for the INPUT2 keyword, the resource names you specify for NAME1 and NAME2 keywords must be the same.

For both NAME1 and NAME2, you can use wildcard characters to create a pattern-matching expression that specifies more than one PSB. The following restrictions apply to using wildcard characters:

- For NAME1, you can use wildcard characters only when NAME2 keyword is omitted.
- For NAME2, you can use wildcard characters only when wildcard characters are not used for the NAME1 keyword value.

### OPTION statement

Optional statement.

Use the following keywords to exclude certain DBDGEN or PSBGEN statements and parameters from the scope of comparison. The OPTION statement works the same as the NOCOMP control statement of the DBD/PSB/ACB Compare utility. For detailed information about statements that are not compared, see [“Summary of NOCOMP keyword parameters for source-level compare”](#) on page 175.

- AREA=

- COMPRTN=
- DBDNAME=
- IMSREL=
- KEYLEN=
- LANG=
- LIST=
- METADATA=
- PCBNAME=
- PROCOPT=
- PROCSEQ=
- PROCSEQD=
- PSB\_ACCESS=
- PSB\_PSELOPT=
- PSBNAME=
- RMNAME=
- VERSION=

**YES**

The DBDGEN or PSBGEN statements and parameters are compared. The default value is YES.

**NO**

The DBDGEN or PSBGEN statements and parameters are not compared.

**REFER\_PSB=**

Specifies whether the utility skips the process to obtain the DBD VERSION or EXIT parameter value of an ACB for a DEDB or MSDB from a PSB-type ACB that references the ACB.

When the utility compares an ACB for a DEDB or MSDB, a PSB-type ACB that references the ACB is used for obtaining the DBD VERSION or EXIT parameter value. When the target library has many members, it can be time-consuming to obtain these values. You can specify whether the utility skips this process to obtain these values. When the utility skips this process to obtain these parameter values, warning message FABL0054W is issued in the Error and Warning Messages report.

**YES**

The utility supplies and compares DBD VERSION or EXIT parameter value of an ACB for a DEDB or MSDB from a PSB-type ACB that references the ACB. This is the default value.

**NO**

The utility does not supply these values from a PSB-type ACB.

**REPORT statement**

Optional statement.

**COMPARE\_LISTING=**

Specifies whether to print the Compare Listing.

**YES**

Prints the Compare Listing even if the utility detects no difference. This is the default value.

**NO**

Does not print the Compare Listing even if the utility detects differences.

**YES\_ONLY\_DIFF**

Prints the Compare Listing only when the utility detects differences.

**END statement**

Optional statement. Use this statement to indicate the end of the control statements.

## 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:

Purpose	Coding example
Compare all DBD-type ACBs in the IMS directory with those in the ACB libraries	DBD NAME1=*, NAME2=*
Compare DBD-type ACBs that have a name that begins with the letter H in the IMS directory with those in the ACB libraries	DBD NAME1=H*, NAME2=H*
Compare PSB-type ACBs that have a name that begins with the letters ABC, have any letters as the fourth character, and contain 001 as the fifth to seventh characters in the IMS directory with those in the ACB libraries	PSB NAME1=ABC%001, NAME2=ABC%001

## Control statements for the convert function

Use the following information to prepare control statements for the convert function of the Catalog Manager utility.

Subsections:

- [“Syntax rules” on page 298](#)
- [“Control statement example” on page 298](#)
- [“Statements, keywords, and parameters” on page 299](#)
- [“Tips for using wildcard characters” on page 301](#)

### 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
```

### Control statement example

The following figure shows an example of the FABXCIN control statements to convert a DBD member and a PSB member in the IMS directory.

```
//FABXCIN DD *  
PROC FUNC=DECODE, INPUT=DIRECTORY_ACTIVE  
*  
OPTION COMMENT=YES, COMPRESS=YES  
DBD NAME=DBD@001  
PSB NAME=PSB@00*  
/*
```

Figure 146. Control statement example (convert function)

## Statements, keywords, and parameters

### PROC statement

Required statement. This statement must be coded on the first line.

To invoke the convert function, specify: FUNC=DECODE

The utility converts DBD and PSB control blocks in the IMS directory to IMS DBDGEN and IMS PSBGEN control statements.

**Note:** Abbreviations are shown in parentheses.

### INPUT=

Specify either of the following values to indicate whether to use active ACBs or staging ACBs in the IMS directory.

#### **DIRECTORY\_ACTIVE (DIR\_ACT)**

Specifies to convert active ACBs in the IMS directory data sets.

#### **DIRECTORY\_STAGING (DIR\_STG)**

Specifies to convert staging ACBs in the staging data set.

### DBD statement

Optional statement. Use this statement to select 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 select specific PSBs. Specify this statement with the following keywords and parameters:

#### **NAME=resource\_name**

Specify a PSB name. You can use wildcard characters to create a pattern-matching expression that specifies more than one PSB.

#### **PCBNAME\_PREFIX=prfx**

Specify, in 1-4 characters, the prefix to use when assigning names to PCBs.

If the utility finds one or more PCBs without PCB names, the utility assigns PCB names to them and uses those names for the PCBNAME parameters of the PCB statement. The naming format is *prfxnnnn*, where *prfx* is the 1-4 characters that the PCBNAME\_PREFIX keyword specifies, and *nnnn* is the PCB number.

The following is an example of the control statement:

```
NAME=psbname , PCBNAME_PREFIX=prfx
```

### OPTION statement

Optional statement. The following options can be specified for converting DBDs and PSBs in the IMS directory.

#### **COMMENT=**

Specifies whether the utility 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. This is the default value.

#### **NO**

The comment lines are not printed.

#### **COMPRESS=**

Specifies whether the decoded DBD or PSB sources are printed in compressed format.

#### **YES**

The decoded sources are printed in compressed format. This is the default value.

**NO**

The decoded sources are printed in noncompressed format.

**FORMAT\_COL10=**

Specifies whether to print the decoded DBDGEN or PSBGEN macro statements starting at column 10. The utility prints one parameter per line, which starts at column 16. When the statement name is longer than 6 characters, one blank is placed between the DBDGEN or PSBGEN macro statements and the parameter that follows.

When both COMPRESS=YES and FORMAT\_COL10=YES are specified, the COMPRESS parameter is ignored and the decoded source is formatted in the above format.

**YES**

The decoded DBDGEN or PSBGEN macro statements start at column 10.

**NO**

The decoded source is printed in the default format. FORMAT\_COL10=NO is the default value.

**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. This is the default value.

**PGM\_COBOL=**

Specifies whether to print LABG=COBOL or LABG=ASSEM in the decoded IMS PSBGEN control statements.

**YES**

Prints LANG=COBOL in the IMS PSBGEN control statements.

**NO**

Prints LANG=ASSEM in the IMS PSBGEN control statements. This is the default value.

**REFER\_PSB=**

Specifies whether the utility skips the process to obtain the DBD VERSION or EXIT parameter value of an ACB for a DEDB or MSDB from a PSB-type ACB that references the ACB. When decoding an ACB for a DEDB or MSDB, a PSB-type ACB that references the ACB is used for obtaining the DBD VERSION or EXIT parameter value. When the target library has many members, it can be time-consuming to obtain these values. You can specify whether the utility skips this process to obtain these values. When the utility skips this process to obtain these parameter values, warning message FABN0077W is issued in both the decoded DBD source and the Error and Warning Messages report.

**YES**

The utility supplies DBD VERSION or EXIT parameter value of an ACB for a DEDB or MSDB from a PSB-type ACB that references the ACB. This is the default value.

**NO**

The utility does not supply these values from a PSB-type ACB.

**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. This is the default value.

**END statement**

Optional statement. Use this statement to indicate the end of the control statements.

## 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:

Purpose	Coding example
Convert all DBD-type ACBs	DBD NAME=*
Convert DBD-type ACBs that have a name that begins with the letter H	DBD NAME=H*
Convert PSB-type ACBs that have a name that begins with the letters ABC, have any letter as the fourth characters, and contain 001 as the fifth to seventh characters	PSB NAME=ABC%001

## Control statements for the map function

Use the following information to prepare control statements for the map function of the Catalog Manager utility.

Subsections:

- [“Syntax rules” on page 301](#)
- [“Control statement example” on page 301](#)
- [“Statements, keywords, and parameters” on page 301](#)
- [“Tips for using wildcard characters” on page 303](#)

### 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
```

### Control statement example

The following figure shows an example of the FABXCIN control statements to generate maps and reports.

```
//FABXCIN DD *  
PROC FUNCTION=MAP,INPUT=CATALOG_DB,INSTANCE=ACTIVE  
REPORT MAP_REPORT=YES  
DBD NAME=TESTDB1  
END  
/*
```

Figure 147. Control statement example (map function)

### Statements, keywords, and parameters

#### PROC statement

Required statement. This statement must be coded on the first line.

To invoke the map function, specify: FUNC=MAP

**INPUT=**

Specify this keyword with the following value:

**CATALOG\_DB(CAT\_DB)**

Specifies to generate maps and reports of the IMS catalog database.

**INSTANCE=**

Specify either of the following values to identify a process instance of the DBD or PSB resource in the IMS catalog database. IMS catalog database has Active or Pending timestamp in its resource header segment. The utility uses this timestamp to process the DBD or PSB instance when you specify ACTIVE or PENDING for the INSTANCE statement. When the IMS catalog database does not have Active or Pending timestamp, the utility uses a timestamp from the IMS directory or ACB library to identify the Active or Pending instance of the IMS catalog database. The library that the utility uses depends on the ACBMGMT= parameter in the DFSDFxxx PROCLIB member. The utility uses the IMS directory when ACBMGMT=CATALOG is specified and the utility uses the ACB library when ACBMGMT=ACBLIB is specified.

**ACTIVE**

Specifies to process the active instance of the DBD or PSB in the IMS catalog database.

**PENDING**

Specifies to process the pending instance of the DBD or PSB in the IMS catalog database.

**INSTANCE\_DBVER=**

Specify the numeric value from 1 to 2147483647 that identifies a specific version of a DBD when multiple DBDs are used by application programs to access the same database. When this statement is omitted, 0 is used to process the instance. This keyword cannot be specified with the INSTANCE= keyword.

**INSTANCE\_GENDATE=**

Specify this keyword with *mm/dd/yyyy* (month/day/year).

This keyword must be specified with the INSTANCE\_GENTIME keyword. This keyword cannot be specified with the INSTANCE= or INSTANCE\_TIMESTAMP keyword.

**INSTANCE\_GENTIME=**

Specify this keyword with *hh:mm:ss.th* (hour/minutes/seconds).

This keyword must be specified after the INSTANCE\_GENDATE keyword. This keyword cannot be specified with the INSTANCE= or INSTANCE\_TIMESTAMP keyword.

**INSTANCE\_TIMESTAMP=**

Specify this keyword with *yydddhhmmssth* (Example: Specify 2306520132435 for 03/06/2023 20:13:24.35)

This keyword cannot be specified with the INSTANCE= keyword or the INSTANCE\_DBVER, INSTANCE\_GENDATE, and INSTANCE\_GENTIME keywords.

**REPORT statement**

Optional statement. Use this statement to select the output.

**MAP\_REPORT=**

Specify either of the following values:

**YES**

Specifies to output all of the reports.

**NO**

Specifies to output only the maps.

**DBD statement**

Optional statement. Use this statement to select the DBD resource. Specify this statement with the following keywords and parameters, separated by commas.

**NAME=resource\_name**

Specify a DBD name to process. You can use wildcard characters to create a pattern-matching expression that specifies more than one DBD.



**INSTANCE=ACTIVE | PENDING**

**INSTANCE\_DBVER= *db version number***

**INSTANCE\_GENDATE=*mm/dd/yyyy***

**INSTANCE\_GENTIME=*hh:mm:ss.th***

**INSTANCE\_TIMESTAMP=*yydddhhmmssth***

Specify these keywords and parameters to identify a process instance of the specific DBD resource with the NAME= keyword. This keyword can overwrite the INSTANCE or INSTANCE\_DBVER, INSTANCE\_GENDATE, INSTANCE\_GENTIME, or INSTANCE\_TIMESTAMP keyword in the PROC statement.

**Note:** INSTANCE, "INSTANCE\_DBVER, INSTANCE\_GENDATE and INSTANCE\_GENTIME", and INSTANCE\_TIMESTAMP cannot be specified at the same time.

### PSB statement

Optional statement. Use this statement to select the PSB resource. Specify this statement with the following keywords and parameters, separated by commas.

**NAME=*resource\_name***

Specify a PSB name to process. You can use wildcard characters to create a pattern-matching expression that specifies more than one PSB.

**INSTANCE=ACTIVE | PENDING**

**INSTANCE\_DBVER= *db version number***

**INSTANCE\_GENDATE=*mm/dd/yyyy***

**INSTANCE\_GENTIME=*hh:mm:ss.th***

**INSTANCE\_TIMESTAMP=*yydddhhmmssth***

Specify these keywords and parameters to identify a process instance of the specific PSB resource with the NAME= keyword. This keyword can overwrite the INSTANCE or INSTANCE\_DBVER, INSTANCE\_GENDATE, INSTANCE\_GENTIME, or INSTANCE\_TIMESTAMP keyword in the PROC statement.

**Note:** INSTANCE, "INSTANCE\_DBVER, INSTANCE\_GENDATE and INSTANCE\_GENTIME", and INSTANCE\_TIMESTAMP cannot be specified at the same time.

For example, the following control statements are for processing DBD DBDHDAM in the IMS catalog database where the active instance is processed.

```
PROC FUNC=MAP, INPUT=CATALOG_DB, INSTANCE=ACTIVE
REPORT MAP_REPORT=YES
DBD NAME=DBDHDAM
```

### 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:

Purpose	Coding example
Process all DBDs	DBD NAME=*
Process all DBDs that have a name that begins with the letter H	DBD NAME=H*

Purpose	Coding example
Process 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	PSB NAME=ABC%001

## JCL examples for the Catalog Manager utility

The following topics provide JCL examples for running the Catalog Manager utility.

### Example: Validating DBDs and PSBs

The figure in this topic shows a JCL example for validating DBDs and PSBs.

```
//CATMANJ JOB . . . . .
//STEP EXEC PGM=FABXCATM
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMS15.SDFSRESL,DISP=SHR
// DD DSN=ITB.SGLXLOAD,DISP=SHR
//DFSRESLB DD DSN=IMS15.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
IOBF=(2048,4,N,N)
IOBF=(4096,4,N,N)
IOBF=(8192,4,N,N)
IOBF=(32000,4,N,N)
/*
//PROCLIB DD DSN=IMS15.PROCLIB
//RECON1 DD DSN=PROD.RECON1,DISP=SHR
//RECON2 DD DSN=PROD.RECON2,DISP=SHR
/*
//ACBLIB DD DSN=PROD.ACBLIB
//FABXPPRM DD *
DLI,FABXCATM,DFSCPL00,,,,,,,,,Y,N,,,,,,,,,DFSDF=CAT
//FABXCIN DD *
PROC FUNC=CHECK_GENTIME
REPORT TIMESTAMP=FORMAT1
DBD NAME=*
PSB NAME=*
END
/*
```

Figure 148. Example of validating DBDs and PSBs

### Example: Comparing IMS control blocks

The figures in this topic show JCL examples for the compare function.

The following example is for comparing an active ACB in the IMS directory with a staging ACB in the IMS directory staging data set.

```

//CATMANJ JOB .....
//STEP EXEC PGM=FABXCATM
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMS15.SDFSRESL,DISP=SHR
// DD DSN=ITB.SGLXLOAD,DISP=SHR
//DFSRESLB DD DSN=IMS15.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
IOBF=(2048,4,N,N)
IOBF=(4096,4,N,N)
IOBF=(8192,4,N,N)
IOBF=(32000,4,N,N)
/*
//PROCLIB DD DSN=IMS15.PROCLIB
//RECON1 DD DSN=PROD.RECON1,DISP=SHR
//RECON2 DD DSN=PROD.RECON2,DISP=SHR
/*
//FABXCRP0 DD SYSOUT=*
//FABXCRP1 DD SYSOUT=*
//FABXCRP2 DD SYSOUT=*
//FABXPPRM DD *
DLI,FABXCATM,DFSCPL00,,,,,,,,,Y,N,,,,,,,,,DFSDF=CAT
//FABXCIN DD *
PROC FUNC=COMPARE,INPUT1=DIRECTORY_ACTIVE,INPUT2=DIRECTORY_STAGING
OPTION METADATA=NO,VERSION=NO
REPORT COMPARE_LISTING=YES
DBD NAME1=DBDHDAM,NAME2=DBDHDM2
PSB NAME1=*
END
/*

```

Figure 149. Example of comparing an active ACB in the IMS directory with a staging ACB in the IMS directory staging data set

The following example is for comparing a DBD and multiple PSBs between the IMS directory and the ACB library.

```

//CATMANJ JOB .....
//STEP EXEC PGM=FABXCATM
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMS15.SDFSRESL,DISP=SHR
// DD DSN=ITB.SGLXLOAD,DISP=SHR
//DFSRESLB DD DSN=IMS15.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
IOBF=(2048,4,N,N)
IOBF=(4096,4,N,N)
IOBF=(8192,4,N,N)
IOBF=(32000,4,N,N)
/*
//PROCLIB DD DSN=IMS15.PROCLIB
//RECON1 DD DSN=PROD.RECON1,DISP=SHR
//RECON2 DD DSN=PROD.RECON2,DISP=SHR
/*
//ACBLIB DD DSN=PROD.ACBLIB
//FABXCRP0 DD SYSOUT=*
//FABXCRP1 DD SYSOUT=*
//FABXCRP2 DD SYSOUT=*
//FABXPPRM DD *
DLI,FABXCATM,DFSCPL00,,,,,,,,,Y,N,,,,,,,,,DFSDF=CAT
//FABXCIN DD *
PROC FUNC=COMPARE,INPUT1=DIRECTORY_ACTIVE,INPUT2=ACBLIB
OPTION METADATA=NO
REPORT COMPARE_LISTING=YES_ONLY_DIFF
DBD NAME1=DBDHDAM,NAME2=DBDHDM2
PSB NAME1=*
END
/*

```

Figure 150. Example of comparing active ACBs in the IMS directory with those in the ACB library

The following example is for comparing a staging ACB in the IMS directory staging data set with a DBD in the DBD library.

```

//CATMANJ JOB .....
//STEP EXEC PGM=FABXCATM
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMS15.SDFSRESL,DISP=SHR
// DD DSN=ITB.SGLXLOAD,DISP=SHR
//DFSRESLB DD DSN=IMS15.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
IOBF=(2048,4,N,N)
IOBF=(4096,4,N,N)
IOBF=(8192,4,N,N)
IOBF=(32000,4,N,N)
/*
//PROCLIB DD DSN=IMS15.PROCLIB
//RECON1 DD DSN=PROD.RECON1,DISP=SHR
//RECON2 DD DSN=PROD.RECON2,DISP=SHR
/*
//DBDLIB DD DSN=PROD.DBDLIB
//FABXCRP0 DD SYSOUT=*
//FABXCRP1 DD SYSOUT=*
//FABXCRP2 DD SYSOUT=*
//FABXPPRM DD *
DLI,FABXCATM,DFSCPL00,,,,,,,,,Y,N,,,,,,,,,DFSDF=CAT
//FABXCIN DD *
PROC FUNC=COMPARE,INPUT1=DIRECTORY_STAGING,INPUT2=DBDLIB
OPTION VERSION=NO,RMNAME=NO
REPORT COMPARE_LISTING=YES_ONLY_DIFF
DBD NAME1=DBDHDAM
END
/*

```

Figure 151. Example of comparing a staging ACB in the IMS directory staging data set with a DBD in the DBD library

## Example: Converting IMS control blocks to control statements

The figure in this topic shows a JCL example for converting the runtime ACBs for databases and program views control blocks in the IMS directory to IMS DBDGEN and IMS PSBGEN control statements.

```
//CATMANJ JOB .....
//STEP EXEC PGM=FABXCATM
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
// DD DSN=IMS15.SDFSRESL,DISP=SHR
// DD DSN=ITB.SGLXLOAD,DISP=SHR
//DFSRESLB DD DSN=IMS15.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
IOBF=(2048,4,N,N)
IOBF=(4096,4,N,N)
IOBF=(8192,4,N,N)
IOBF=(32000,4,N,N)
/*
//PROCLIB DD DSN=IMS15.PROCLIB
//RECON1 DD DSN=PROD.RECON1,DISP=SHR
//RECON2 DD DSN=PROD.RECON2,DISP=SHR
//*
//FABXCRP0 DD SYSOUT=*
//FABXCRP1 DD SYSOUT=*
//FABXCSRC DD SYSOUT=*
//FABXPPRM DD *
DLI,FABXCATM,DFSCPL00,,,,,,,,,,,,,Y,N,,,,,,,,,,,,,DFSDF=CAT
//FABXCIN DD *
PROC FUNC=DECODE,INPUT=DIRCTORY_STAGING
OPTION COMMENT=YES,COMPRESS=YES
DBD NAME=*
PSB NAME=*
END
/*
```

Figure 152. Example of converting ACBs in the IMS directory staging data set

## Example: Generating maps and reports

The figures in this topic show JCL examples for the map function.

### Example: Generating a DBD map and a DBD report of an active DBD

The following figure shows a JCL example for the map function.

The following example is for generating a map and a report of an active DBD in the IMS catalog database.

```

//CATMANJ      JOB .....
//STEP        EXEC PGM=FABXCATM
//STEPLIB     DD DSN=HPS.SHPSLMD0,DISP=SHR
//            DD DSN=IMS15.SDFSRESL,DISP=SHR
//            DD DSN=ITB.SGLXLOAD,DISP=SHR
//DFSRESLB    DD DSN=IMS15.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
IOBF=(2048,4,N,N)
IOBF=(4096,4,N,N)
IOBF=(8192,4,N,N)
IOBF=(32000,4,N,N)
/*
//PROCLIB    DD DSN=IMS15.PROCLIB
//RECON1     DD DSN=PROD.RECON1,DISP=SHR
//RECON2     DD DSN=PROD.RECON2,DISP=SHR
//*
//FABXCRP0   DD SYSOUT=*
//FABXCRP1   DD SYSOUT=*
//FABXCRP2   DD SYSOUT=*
//FABXPPRM   DD *
DLI,FABXCATM,DFSCPL00,,,,,,,,,Y,N,,,,,,,,,DFSDF=CAT
//FABXCIN    DD *
  PROC  FUNC=MAP,INPUT=CATALOG_DB,INSTANCE=ACTIVE
  REPORT MAP_REPORT=YES
  DBD  NAME=DBDHDAM
  END
/*

```

Figure 153. Example of generating a DBD map and a DBD report for an active DBD instance in the IMS catalog database

### Example: Generating a DBD map of the specified DBD

The following figure shows a JCL example for the map function.

The following example is for generating a map of the specified DBD with the `INSTANCE_DBVER`, `INSTANCE_GENDATE`, and `INSTANCE_GENTIME` keywords for an instance in the IMS catalog database without a DBD report.

```

//CATMANJ      JOB .....
//STEP        EXEC PGM=FABXCATM
//STEPLIB     DD DSN=HPS.SHPSLMD0,DISP=SHR
//           DD DSN=IMS15.SDFSRESL,DISP=SHR
//           DD DSN=ITB.SGLXLOAD,DISP=SHR
//DFSRESLB   DD DSN=IMS15.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
IOBF=(2048,4,N,N)
IOBF=(4096,4,N,N)
IOBF=(8192,4,N,N)
IOBF=(32000,4,N,N)
/*
//PROCLIB   DD DSN=IMS15.PROCLIB
//RECON1    DD DSN=PROD.RECON1,DISP=SHR
//RECON2    DD DSN=PROD.RECON2,DISP=SHR
//*
//FABXCRP0  DD SYSOUT=*
//FABXCRP1  DD SYSOUT=*
//FABXCRP2  DD SYSOUT=*
//FABXPPRM  DD *
DLI,FABXCATM,DFSCPL00,,,,,,,,,Y,N,,,,,,,,,DFSDF=CAT
//FABXCIN  DD *
      PROC  FUNC=MAP,INPUT=CATALOG_DB
      REPORT MAP_REPORT=NO
      DBD  NAME=DBDHDAM,
           INSTANCE_DBVER=0,
           INSTANCE_GENDATE=03/24/2023,
           INSTANCE_GENTIME=21:04:22.15
      END
/*

```

Figure 154. Example of generating a DBD map

## Example: Generating PSB maps, PSB reports, and a PSB summary report

The following figure shows a JCL example for the map function.

The following example is for generating maps and reports of the specified PSB with the `TIMESTAMP` parameter in the IMS catalog database.



```

//CATMANJ      JOB .....
//STEP        EXEC PGM=FABXCATM
//STEPLIB     DD DSN=HPS.SHPSLMD0,DISP=SHR
//           DD DSN=IMS15.SDFSRESL,DISP=SHR
//           DD DSN=ITB.SGLXLOAD,DISP=SHR
//DFSRESLB    DD DSN=IMS15.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
IOBF=(2048,4,N,N)
IOBF=(4096,4,N,N)
IOBF=(8192,4,N,N)
IOBF=(32000,4,N,N)
/*
//PROCLIB    DD DSN=IMS15.PROCLIB
//RECON1     DD DSN=PROD.RECON1,DISP=SHR
//RECON2     DD DSN=PROD.RECON2,DISP=SHR
/*
//FABXCRP0   DD SYSOUT=*
//FABXCRP1   DD SYSOUT=*
//FABXCRP2   DD SYSOUT=*
//FABXPPRM   DD *
DLI,FABXCATM,DFSCPL00,,,,,,,,,Y,N,,,,,,,,,DFSDF=CAT
//FABXCIN    DD *
PROC FUNC=MAP,INPUT=CATALOG_DB
REPORT MAP_REPORT=YES
PSB NAME=PSB001,INSTANCE_TIMESTAMP=2306521040977
END
/*

```

Figure 155. Example of generating PSB maps, PSB reports, and a PSB summary report for an instance in the IMS catalog database

## Output from the Catalog Manager utility

Output from the Catalog Manager utility consists of the FABXCRP0 data set, FABXCRP1 data set, FABXCRP2 data set, and FABXCSRC data set. The data sets used depend on the function.

### Output from the validate function

Output from the Catalog Manager utility for validating DBDs or PSBs consists of the FABXCRP0 data set and the FABXCRP1 data set.

#### FABXCRP0 data set (Validate function)

The FABXCRP0 data set contains the Control Statement report, which shows the echo of the FABXCIN control statements and the selected runtime options.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER                "CONTROL STATEMENT REPORT"                PAGE: 1
5655-U08                                                         DATE: 04/05/2021  TIME: 20.04.44                FABXCATM - V2.R2

"CONTROL STATEMENTS"
0.....1.....2.....3.....4.....5.....6.....7.....8
1234567890123456789012345678901234567890123456789012345678901234567890
PROC FUNC=CHECK_GENTIME
REPORT TIMESTAMP=FORMAT1
* Check all DBD-type & PSB-type resources.
DBD NAME=*
PSB NAME=*
END

"RUNTIME OPTIONS"
STATEMENT      KEYWORD      RUNTIME OPTIONS FOR THIS STEP
-----
PROC           FUNC         CHECK_GENTIME

OPTIONAL STATEMENTS
STATEMENT      KEYWORD      RUNTIME OPTIONS FOR THIS STEP
-----
REPORT        TIMESTAMP    FORMAT1

```

Figure 156. Example of the Control Statement report

## FABXCRP1 data set (Validate function)

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” on page 312](#)
- [“IMS Catalog Environment report” on page 312](#)
- [“IMS Catalog Validation report” on page 313](#)

## FABXPPRM Echo Back report

This report contains an echo of the FABXPPRM parameters.

```
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "FABXPPRM ECHO BACK"          PAGE: 1
5655-U08          DATE: 04/05/2021  TIME: 20.04.44          FABXCATM - V2.R2

"FABXPPRM STATEMENT"

0.....1.....2.....3.....4.....5.....6.....7.....8
123456789012345678901234567890123456789012345678901234567890
DLI,FABXCAPL,DFSCP000,,,,,,,,,N,N,,,,,,,,,DFSDF=RGN
```

Figure 157. Example of the FABXPPRM Echo Back report

## IMS Catalog Environment report

This report contains environment information about the IMS system and the IMS catalog.

```
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "IMS CATALOG ENVIRONMENT REPORT"          PAGE: 1
5655-U08          DATE: 04/05/2021  TIME: 20.04.44          FABXCATM - V2.R2

IMS ENVIRONMENT

IMS ID          : SYS1
IMS VERSION     : 15.01.00

IMS CATALOG ENVIRONMENT

DFSDF MEMBER NAME : DFSDFRGN
DFS3CDX0 ROUTINE  : NO
ACB MANAGEMENT   : ACBLIB

ALIAS OF CATALOG DB : DFSC
CATALOG HLQ       : IMS.CATALOG
IMS DIRECTORY HLQ  : IMS.CATALOG
```

Figure 158. Example of the IMS Catalog Environment report

This report contains the following fields:

### IMS ID

IMS ID.

### IMS VERSION

IMS version. The utility, which can retrieve the IMS installed level of 15.2 or later, reports the IMS installed level instead of the IMS base level.

### 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 OF CATALOG DB

The alias name of the IMS catalog database.

## CATALOG HLQ

The high-level qualifier (HLQ) of the IMS catalog.

## IMS DIRECTORY HLQ

The high-level qualifier of the IMS directory data sets. This field is blank if the IMS directory was not referred to in the job.

## IMS Catalog Validation report

This report contains validation results for DBDs and PSBs. If ACBs were loaded from ACB libraries, the report contains validation results for the DBDs and PSBs in the ACB libraries and the IMS catalog. 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, and the IMS directory.

Report examples:

- [Example of the IMS Catalog Validation report \(ACB library and IMS catalog\)](#)
- [Example of the IMS Catalog Validation report \(IMS directory, IMS catalog, and ACB library\)](#)
- [Example of the IMS Catalog Validation report \(IMS directory and IMS catalog\)](#)

The following figure shows an example of the report when the IMS management of ACBs is not enabled.

```
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "IMS CATALOG VALIDATION REPORT"          PAGE: 1
5655-U08          DATE: 06/13/2021  TIME: 01.25.31          FABXCATM - V2.R2

DBD
---
NAME      VALIDATION |----- ACBLIB -----|----- CATALOG -----|
RESULT    DB VERSION ACBGEN  TIMESTAMP      DB VERSION  TIMESTAMP
-----
DBDXD70A  VALID          0000000001  2021/03/03  20:59:27.31  0000000001  2021/03/03  20:59:27.31
DBDXE70A  VALID          0000004321  2021/03/03  20:59:27.31  0000004321  2021/03/03  20:59:27.31
DBDXH70A  VALID          0000000020  2021/03/03  20:59:27.31  0000000020  2021/03/03  20:59:27.31
DBDXI03A  INVALID        0000000000  2021/03/05  10:21:00.05  0000000000  2021/03/03  20:59:27.31

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "IMS CATALOG VALIDATION REPORT"          PAGE: 2
5655-U08          DATE: 06/13/2021  TIME: 01.25.31          FABXCATM - V2.R2

PSB
---
NAME      VALIDATION |----- ACBLIB -----|----- CATALOG -----|
RESULT    ACBGEN  TIMESTAMP      TIMESTAMP
-----
PSBX010  VALID          2021/03/03  20:59:27.31  2021/03/03  20:59:27.31
PSBX013  INVALID        2021/03/05  10:21:00.05  2021/03/03  20:59:27.31
PSBX020  VALID          2021/03/03  20:59:27.31  2021/03/03  20:59:27.31
PSBX030  VALID          2021/03/03  20:59:27.31  2021/03/03  20:59:27.31
```

Figure 159. Example of the IMS Catalog Validation report (ACB library and IMS catalog)

The following figure shows an example of the report when the IMS management of ACBs is enabled. This example contains result of validating DBDs and PSBs in ACB libraries, IMS catalog, and IMS directory.

DBD		----									
NAME	VALIDATION RESULT	STATUS	DIRECTORY TIMESTAMP	DB VERSION	CATALOG TIMESTAMP	DB VERSION	ACBLIB ACBGEN	TIMESTAMP	-----		
DBD#D01A	VALID	STAGING	2021/03/03 21:00:34.51	0000000000	2021/03/03 21:00:34.51	0000000000	2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
DBD#D03A	INVALID	STAGING	2021/03/03 21:00:34.51	(DBD NOT EXIST)		0000000000	2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
DBD#D60A	INVALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	0000000000	2021/02/27 19:37:00.16	(DBD NOT EXIST)					
DBD#D70A	VALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	0000000001	2021/02/27 19:37:00.16	2021/03/03 21:00:34.51	0000000001	2021/03/03 21:00:34.51			
DBD#D80A	INVALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	0000000000	2021/02/27 19:37:00.16	(DBD NOT EXIST)		2021/03/03 21:00:34.51			
DBD#H01A	INVALID	STAGING	2021/03/03 21:00:34.51	(INSTANCE NOT EXIST)		0000000000	2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
DBD#H02A	INVALID	STAGING	2021/03/03 21:00:34.51	(DBD NOT EXIST)		(DBD NOT EXIST)					
DBD#H60A	INVALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	0000000000	2021/02/27 19:37:00.16	(DBD NOT EXIST)					
DBD#H70A	VALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	0000004096	2021/02/27 19:37:00.16	2021/03/03 21:00:34.51	0000004096	2021/03/03 21:00:34.51			
DBD#H80A	INVALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	0000000000	2021/02/27 19:37:00.16	(INSTANCE NOT EXIST)		2021/03/03 21:00:34.51			
DBD#X01A	VALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	0000000000	2021/02/27 19:37:00.16	2021/03/03 21:00:34.51	0000000000	2021/03/03 21:00:34.51			
DBD#X02A	VALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	0000000000	2021/02/27 19:37:00.16	2021/03/03 21:00:34.51	0000000000	2021/03/03 21:00:34.51			
DBD@G20V	INVALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	(INSTANCE NOT EXIST)		0000000000	2021/03/03 21:00:34.51	(GSAM)			
DBD@X03A	INVALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	(INSTANCE NOT EXIST)		0000000000	*2021/02/27 19:37:00.16				

Figure 160. Example of the IMS Catalog Validation report (IMS directory, IMS catalog, and ACB library) (Part 1 of 2)

PSB		----									
NAME	VALIDATION RESULT	STATUS	DIRECTORY TIMESTAMP	DB VERSION	CATALOG TIMESTAMP	DB VERSION	ACBLIB ACBGEN	TIMESTAMP	-----		
DFSCPL00	VALID	ACTIVE	2021/02/27 19:37:01.86		2021/02/27 19:37:01.86		2021/02/27 19:37:01.86	2021/02/27 19:37:01.86			
DFSCP000	VALID	ACTIVE	2021/02/27 19:37:01.86		2021/02/27 19:37:01.86		2021/02/27 19:37:01.86	2021/02/27 19:37:01.86			
DFSCP001	VALID	ACTIVE	2021/02/27 19:37:01.86		2021/02/27 19:37:01.86		2021/02/27 19:37:01.86	2021/02/27 19:37:01.86			
MAXPSB01	VALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51		2021/02/27 19:37:00.16	2021/03/03 21:00:34.51		2021/03/03 21:00:34.51			
MAXPSB02	VALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51		2021/02/27 19:37:00.16	2021/03/03 21:00:34.51		2021/03/03 21:00:34.51			
PSB#001	INVALID	-	(PSB NOT EXIST)		2021/03/03 21:00:34.51		2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
PSB#002	VALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51		2021/02/27 19:37:00.16	2021/03/03 21:00:34.51		2021/03/03 21:00:34.51			
PSB#009	VALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51		2021/02/27 19:37:00.16	2021/03/03 21:00:34.51		2021/03/03 21:00:34.51			
PSB#031	VALID	STAGING	2021/03/03 21:00:34.51		2021/03/03 21:00:34.51		2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
PSB#032	VALID	STAGING	2021/03/03 21:00:34.51		2021/03/03 21:00:34.51		2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
PSB#033	VALID	STAGING	2021/03/03 21:00:34.51		2021/03/03 21:00:34.51		2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
PSB#600	INVALID	STAGING	2021/03/03 21:00:34.51	(PSB NOT EXIST)			2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
PSB#700	INVALID	STAGING	2021/03/03 21:00:34.51	(PSB NOT EXIST)			2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
PSB#800	INVALID	STAGING	2021/03/03 21:00:34.51	(PSB NOT EXIST)			2021/03/03 21:00:34.51	2021/03/03 21:00:34.51			
PSB@G10B	INVALID	STAGING	2021/03/03 21:00:34.51	(PSB NOT EXIST)			(PSB NOT EXIST)				
PSB@G10V	VALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51		2021/02/27 19:37:00.16	2021/03/03 21:00:34.51		2021/03/03 21:00:34.51			
PSB@600	INVALID	ACTIVE STAGING	2021/02/27 19:37:00.16 2021/03/03 21:00:34.51	(INSTANCE NOT EXIST)			*2021/02/27 19:37:00.16				

Figure 161. Example of the IMS Catalog Validation report (IMS directory, IMS catalog, and ACB library) (Part 2 of 2)

The following figure shows an example of the report when the IMS management of ACBs is enabled. This example contains result of validating DBDs and PSBs in IMS catalog, and IMS directory.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "IMS CATALOG VALIDATION REPORT"
5655-U08          DATE: 06/15/2021  TIME: 02.09.27          PAGE: 1
                                                           FABXCATM - V2.R2

DBD
---
NAME      VALIDATION |----- DIRECTORY -----|----- CATALOG -----|
          RESULT    |-----|          |-----|          |
          STATUS   |TIMESTAMP|          |TIMESTAMP|
          |-----|          |-----|          |
DBD#D22A  VALID
          |-----|          |-----|          |
          ACTIVE   |2021/03/05 01:01:33.65 0000000001 2021/03/05 01:01:33.65
          STAGING  |2021/03/09 21:00:34.51 0000000002 2021/03/09 21:00:34.51
DBD#H35A  VALID
          |-----|          |-----|          |
          ACTIVE   |2021/03/05 01:01:33.65 0000000020 2021/03/05 01:01:33.65
DBD#D85A  INVALID
          |-----|          |-----|          |
          ACTIVE   |2021/03/05 01:01:33.65 (INSTANCE NOT EXIST)
DBD@D11A  VALID
          |-----|          |-----|          |
          ACTIVE   |2021/03/05 01:01:33.65 000001234 2021/03/05 01:01:33.65
          STAGING  |2021/03/09 21:00:34.51 000001234 2021/03/09 21:00:34.51
DBD@E33A  VALID
          |-----|          |-----|          |
          ACTIVE   |2021/03/05 01:01:33.65 000004321 2021/03/05 01:01:33.65
DBD@H60A  INVALID
          |-----|          |-----|          |
          ACTIVE   |2021/02/27 19:37:00.16 (INSTANCE NOT EXIST)
          STAGING  |2021/03/09 21:00:34.51 0000000000 2021/03/09 21:00:34.51
DBD@H90A  VALID
          |-----|          |-----|          |
          ACTIVE   |2021/03/05 01:01:33.65 000011111 2021/03/05 01:01:33.65
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "IMS CATALOG VALIDATION REPORT"
5655-U08          DATE: 06/15/2021  TIME: 02.09.27          PAGE: 2
                                                           FABXCATM - V2.R2

PSB
---
NAME      VALIDATION |----- DIRECTORY -----|----- CATALOG -----|
          RESULT    |-----|          |-----|          |
          STATUS   |TIMESTAMP|          |TIMESTAMP|
          |-----|          |-----|          |
PSB@015  VALID
          |-----|          |-----|          |
          ACTIVE   |2021/03/05 01:01:33.65          2021/03/05 01:01:33.65
          STAGING  |2021/03/09 21:00:34.51          2021/03/09 21:00:34.51
PSB@025  VALID
          |-----|          |-----|          |
          ACTIVE   |2021/03/05 01:01:33.65          2021/03/05 01:01:33.65
PSB@035  VALID
          |-----|          |-----|          |
          ACTIVE   |2021/03/05 01:01:33.65          2021/03/05 01:01:33.65

```

Figure 162. Example of the IMS Catalog Validation report (IMS directory and IMS catalog)

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**

When the IMS management of ACBs is enabled, the utility reports VALID if an instance in the IMS catalog has the same ACB generation time stamp as the ACB in the IMS directory.

If an ACB library is specified, the following conditions must also be true to be VALID:

- If the member exists in both the active and staging data sets of the IMS directory:
  - The time stamp is the same between the ACB library and the IMS directory staging data set, and an instance with the time stamp exists in the IMS catalog.
  - The database version number is the same between the ACB library and the IMS catalog.
  - An instance that has the same time stamp as the one in the IMS directory active data set exists in the IMS catalog.
- If the member exists in either the active or the staging data sets of the IMS directory:
  - The time stamp is the same between the ACB library and the IMS directory, and an instance with the same time stamp exists in the IMS catalog.
  - The database version number is the same between the ACB library and the IMS catalog.

When the IMS management of ACBs is not enabled, the utility reports the validation result as VALID if both of the following conditions are met:

- The most recent instance in the IMS catalog has the same ACB generation time stamp as the ACB in the ACB library.
- The database version number of the ACB in the ACB library matches the database version number of the instance in the IMS catalog.

**INVALID**

If the resources do not meet the VALID conditions, INVALID is printed.

When a member is reported as INVALID, identify the cause and run the ACB Generation and Catalog Populate utility (DFS3UACB) or the IMS Catalog Populate utility (DFS3PU00) to fix the condition.

**DIRECTORY****STATUS****ACTIVE**

DBD or PSB is in the active ACB data sets of the IMS directory.

**STAGING**

DBD or PSB is in the IMS directory staging data set.

**TIMESTAMP**

ACB generation time stamp of the DBD or PSB in the active ACB data sets of the IMS directory or the IMS directory staging data set.

When the DBD or PSB does not exist, (DBD NOT EXIST) or (PSB NOT EXIST) is printed.

**CATALOG****DB VERSION**

The database version number. 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.

**TIMESTAMP**

ACB generation time stamp of the DBD or PSB in the IMS catalog.

- When the IMS management of ACBs is enabled and the ACB exists in the IMS directory, this field shows the ACB generation time stamp of the DBD or PSB, which is the same as the ACB generation time stamp of the ACB in the IMS directory.

If an instance with the same time stamp does not exist in the IMS catalog, this field shows (INSTANCE NOT EXIST).

If the ACB does not exist in the IMS directory, this field shows the time stamp of the ACB in the ACB library.

- When the IMS management of ACBs is not enabled, it shows the time stamp of the most recent DBD or PSB in the IMS catalog.

(DBD NOT EXIST) or (PSB NOT EXIST) is printed if the utility finds no instance of the DBD or PSB in the IMS catalog.

**ACBLIB****DB VERSION**

The database version number. 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.

**ACBGEN TIMESTAMP**

ACB generation time stamp.

If the time stamp does not match with the time stamp of the ACB in the IMS directory, an asterisk (\*) is printed before the time stamp.

(DBD NOT EXIST) or (PSB NOT EXIST) is printed if the utility finds no ACB in the ACB libraries.

## Output from the compare function

Output from the Catalog Manager utility for comparing IMS control blocks consists of the FABXCRP0 data set, FABXCRP1 data set, and FABXCRP2 data set.

### FABXCRP0 data set (Compare function)

The FABXCRP0 data set contains the Control Statement report, which shows the echo of the FABXCIN control statements and the selected runtime options.

```
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "CONTROL STATEMENT REPORT"          PAGE: 1
5655-U08          DATE: 04/05/2021  TIME: 20.04.44          FABXCATM - V2.R2

"CONTROL STATEMENTS"
0.....1.....2.....3.....4.....5.....6.....7.....8
1234567890123456789012345678901234567890123456789012345678901234567890

  PROC FUNC=COMPARE,INPUT1=DIRECTORY_ACTIVE,INPUT2=ACBLIB
  DBD NAME1=DBD*
  PSB NAME1=PSB*
  END

"RUNTIME OPTIONS"

STATEMENT      KEYWORD      RUNTIME OPTIONS FOR THIS STEP
-----
PROC           FUNC         COMPARE

OPTIONAL STATEMENTS

STATEMENT      KEYWORD      RUNTIME OPTIONS FOR THIS STEP
-----
OPTION         VERSION      YES
               DBDNAME      YES
               PSBNAME      YES
               METADATA   YES
               AREA       YES
               RMNAME     YES
               COMPRTN   YES
               PCBNAME    YES
               KEYLEN     YES
               IMSREL    YES
               LANG       YES
               LIST       YES
               PROCOPT   YES
               PROCSEQ   YES
               PROCSEQD  YES
               PSB_ACCESS YES
               PSB_PSELOPT YES
               REFER_PSB  YES
REPORT        COMPARE_LISTING  YES
```

Figure 163. Example of the Control Statement report

### FABXCRP1 data set (Compare function)

The FABXCRP1 data set contains the FABXPPRM Echo Back report, the IMS Catalog Environment report, the Compare Summary report, and, if errors or warning messages were issued, the Error and Warning Messages report.

Subsections:

- [“FABXPPRM Echo Back report” on page 317](#)
- [“IMS Catalog Environment report” on page 318](#)
- [“Compare Summary report” on page 318](#)
- [“Error and Warning Messages report” on page 319](#)

### FABXPPRM Echo Back report

This report contains an echo of the FABXPPRM parameters.

```
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "FABXPPRM ECHO BACK"          PAGE: 1
5655-U08          DATE: 04/05/2021  TIME: 20.04.44          FABXCATM - V2.R2

"FABXPPRM STATEMENT"
0.....1.....2.....3.....4.....5.....6.....7.....8
1234567890123456789012345678901234567890123456789012345678901234567890
DLI,FABXCAPL,DFSCP000,,,,,,,,,,,,,N,N,,,,,,,,,,,,,DFSDF=RGN
```

Figure 164. Example of the FABXPPRM Echo Back report

## IMS Catalog Environment report

This report contains environment information about the IMS system and the IMS catalog.

```
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "IMS CATALOG ENVIRONMENT REPORT"          PAGE: 1
5655-U08                                                    DATE: 04/05/2021  TIME: 20.04.44          FABXCATM - V2.R2

IMS ENVIRONMENT

IMS ID              : SYS1
IMS VERSION         : 15.01.00

IMS CATALOG ENVIRONMENT

DFSDF MEMBER NAME   : DFSDFRGN
DFS3CDX0 ROUTINE    : NO
ACB MANAGEMENT     : ACBLIB

ALIAS OF CATALOG DB : DFSC
CATALOG HLQ        : IMS.CATALOG
IMS DIRECTORY HLQ  : IMS.CATALOG
```

Figure 165. Example of the IMS Catalog Environment report

This report contains the following fields:

### IMS ID

IMS ID.

### IMS VERSION

IMS version. The utility, which can retrieve the IMS installed level of 15.2 or later, reports the IMS installed level instead of the IMS base level.

### 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 OF CATALOG DB

The alias name of the IMS catalog database.

### CATALOG HLQ

The high-level qualifier (HLQ) of the IMS catalog.

### IMS DIRECTORY HLQ

The high-level qualifier of the IMS directory data sets. This field is blank if the IMS directory was not referred to in the job.

## Compare Summary report

This report shows the comparison summary.



LIBRARY INFORMATION

INPUT1  
IMS DIRECTORY HLQ : IMSVS.IMS15A.DFSCD000  
STATUS : ACTIVE  
INPUT2  
ACBLIB : IMSVS.IMS15A.ACBLIB1

RESOURCE TYPE	RESOURCE NAME1	RESOURCE NAME2	RESULT
DBD	DBD#D01A	DBD#D03A	DIFFERENCE
DBD	DBD#H01A	DBD#H01A	SAME
DBD	DBD#X01A	DBD#X02A	DIFFERENCE
DBD	DBD@G01A	DBD@G01A	FAIL
PSB	PSB#600	PSB#001	DIFFERENCE
PSB	PSB#700	PSB#700	SAME
PSB	PSB#800	PSB#800	SAME
PSB	PSB@001	PSB@001	FAIL
DBD TOTAL	4	1	DIFFERENCE 2 FAIL 1
PSB TOTAL	4	2	DIFFERENCE 1 FAIL 1

Figure 166. Example of the Compare Summary report

This report contains the following fields:

**INPUT1**

The library that is specified for the INPUT1 keyword of the PROC FUNC=COMPARE statement.

**INPUT2**

The library that is specified for the INPUT2 keyword of the PROC FUNC=COMPARE statement.

**RESOURCE TYPE**

The type of the resource compared. DBD or PSB.

**RESOURCE NAME1**

The name of the resource found in the library specified by the INPUT1 keyword.

**RESOURCE NAME2**

The name of the resource found in the library specified by the INPUT2 keyword.

**RESULT**

**SAME**

No difference is found.

**DIFFERENCE**

Difference is found.

**FAIL**

Member is not found in both or one of the specified libraries.

**Error and Warning Messages report**

This report contains error and warning messages. If error or warning messages were issued during the process, those messages are printed to the Error and Warning Messages report. If no error or warning messages were issued, this report contains "No message".

FABX0557W DIFFERENCE FOUND DURING COMPARE ACB=DBD#D01A:DBD#D03A  
FABX0557W DIFFERENCE FOUND DURING COMPARE ACB=DBD#X01A:DBD#X02A  
FABL0047W GSAM DBD DBD@G01A IS NOT COMPARED  
FABX0557W DIFFERENCE FOUND DURING COMPARE ACB=PSB#600 :PSB#001  
FABL0005W NO MEMBER FOUND FOR PSB@001 IN IMS DIRECTORY

Figure 167. Example of the Error and Warning Messages report

## FABXCRP2 data set (Compare function)

The FABXCRP2 data set contains the Compare Listing. This report contains details about the differences detected.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "COMPARE LISTING"          PAGE: 1
5655-U08          DATE: 04/05/2021  TIME: 20.04.44          FABXCATM - V2.R2
NUMBER OF DIFFERENT STATEMENTS
INSERTED : 0
DELETED  : 0
CHANGED  : 4
IMS DIRECTORY HLQ :
IMSVS.IMS15A.DFSCD000
STATUS : ACTIVE
RESOURCE : DBD#X01A
GENERATED : 03/18/2021 21.27
GENERATED IMS : 1510
ACBLIB :
IMSVS.IMS15A.ACBLIB1
RESOURCE : DBD#X02A
ACBGEN : 03/18/2021 21.27
GENERATED IMS : 1510

CHK SOURCE LINES          SOURCE LINES
-----1-----2-----3-----4-----5-----6-
C - DBD          00000001          DBD          00000001
*   NAME=DBD#X01A,          00000002          NAME=DBD#X02A,          00000002
ACCESS=(PSINDEX,VSAM,PROT,DOSCOMP),          00000003          ACCESS=(PSINDEX,VSAM,PROT,DOSCOMP),          00000003
PASSWD=YES,          00000004          PASSWD=YES,          00000004
*   VERSION='1XXXXXX'          00000005          VERSION= 03/05/21 23.33          00000005
C - SEGM          00000006          SEGM          00000006
*   NAME=X01AS001,          00000007          NAME=X02AS001,          00000007
PARENT=0,          00000008          PARENT=0,          00000008
BYTES=64,          00000009          BYTES=64,          00000009
RULES=(LLL, LAST),          00000010          RULES=(LLL, LAST),          00000010
DSGROUP=A          00000011          DSGROUP=A          00000011
C - FIELD          00000012          FIELD          00000012
*   NAME=(X01AFLA,SEQ,U),          00000013          NAME=(X02AFLA,SEQ,U),          00000013
START=1,          00000014          START=1,          00000014
*   BYTES=18,          00000015          BYTES=10,          00000015
*   TYPE=C          00000016          TYPE=X          00000016
C - LCHILD          00000017          LCHILD          00000017
*   NAME=(D01SEG2,DBD#D01A),          00000018          NAME=(D01SEG31,DBD#D01A),          00000018
INDEX=XDF02D01,          00000019          INDEX=XDF02D01,          00000019
*   RKSIZE=10          00000020          RKSIZE=10          00000020
DBDGEN          00000021          DBDGEN          00000021
FINISH          00000022          FINISH          00000022
END          00000023          END          00000023

```

Figure 168. Example of the Compare Listing

This report contains the following fields:

### 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 the DBD or the PSB in the library specified by the INPUT2 keyword.

#### DELETED

The number of statements which were found only in the DBD or the PSB in the library specified by the INPUT1 keyword.

#### CHANGED

The number of statements that exist in both DBDs or PSBs but are different.

### IMS DIRECTORY HLQ

The high-level qualifier HLQ of the IMS directory data sets.

### STATUS

The status of the IMS directory. ACTIVE or STAGING.

### ACBLIB

### DBDLIB

### PSBLIB

The library type. This value is determined from the INPUT2 keyword.

### RESOURCE

The name of the member compared.

### GENERATED

The date and time when the member in the IMS directory was generated.

### ACBGEN

The date and time when the member in the ACB library, DBD library, or PSB library was generated.

## GENERATED IMS

The IMS version and release when the member was generated.

## CHK

The following characters are used to indicate the difference:

### I

A statement is inserted into the DBD or the PSB in the library specified by the INPUT2 keyword.

### D

A statement is deleted from the DBD or the PSB in the library specified by INPUT1 keyword.

### C

A statement in the DBD or the PSB in the library specified by the INPUT1 keyword is different from that in the library specified by the INPUT2= keyword. An asterisk (\*) is shown on the row of each data that is determined to be different.

## SOURCE LINES

The IMS DBDGEN or PSBGEN control statements that were decoded from the DBD or the PSB. The left column shows control statements decoded from the resource found in the library that the INPUT1 keyword specifies, the right column shows control statements decoded from the resource found in the library that the INPUT2 keyword specifies.

## Output from the convert function

Output from the Catalog Manager utility for converting DBDs or PSBs consists of the FABXCRP0 data set, FABXCRP1 data set, and FABXCSRC data set. If you specify the DBDSRC or the PSBSRC data set, the utility generates members containing IMS DBDGEN or PSBGEN control statements in those data sets.

### FABXCRP0 data set (Convert function)

The FABXCRP0 data set contains the Control Statement report, which shows the echo of the FABXCIN control statements and the selected runtime options.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "CONTROL STATEMENT REPORT"
5655-U08                                                    DATE: 04/05/2021  TIME: 20.04.44
                                                            PAGE:          1
                                                            FABXCATM - V2.R2

"CONTROL STATEMENTS"
0.....1.....2.....3.....4.....5.....6.....7.....8
1234567890123456789012345678901234567890123456789012345678901234567890

    PROC FUNC=DECODE,INPUT=DIRECTORY_ACTIVE
    DBD NAME=DBD@003A
    PSB NAME=PSB@003
    END

"RUNTIME OPTIONS"
STATEMENT      KEYWORD      RUNTIME OPTIONS FOR THIS STEP
-----
PROC           FUNC           DECODE

OPTIONAL STATEMENTS
STATEMENT      KEYWORD      RUNTIME OPTIONS FOR THIS STEP
-----
OPTION         PGM_COBOL    NO
               COMMENT    YES
               COMPRESS   YES
               PCB_LABEL   NO
               SENSEG_PROCOPT NO
               REFER_PSB   YES
               FORMAT_COL10 NO
    
```

Figure 169. Example of the Control Statement report

### FABXCRP1 data set (Convert function)

The FABXCRP1 data set contains the FABXPPRM Echo Back report, the IMS Catalog Environment report, and, if errors or warning messages were issued, the Error and Warning Messages report.

Subsections:

- [“FABXPPRM Echo Back report” on page 322](#)
- [“IMS Catalog Environment report” on page 322](#)

- [“Error and Warning Messages report” on page 323](#)

## FABXPPRM Echo Back report

This report contains an echo of the FABXPPRM parameters.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "FABXPPRM ECHO BACK"          PAGE: 1
5655-U08          DATE: 04/05/2021  TIME: 20.04.44          FABXCATM - V2.R2

"FABXPPRM STATEMENT"

0.....1.....2.....3.....4.....5.....6.....7.....8
1234567890123456789012345678901234567890123456789012345678901234567890
DLI,FABXCAPL,DFSCP000,,,,,,,,,,,,,N,N,,,,,,,,,,,,,DFSDF=RGN

```

Figure 170. Example of the FABXPPRM Echo Back report

## IMS Catalog Environment report

This report contains environment information about the IMS system and the IMS catalog.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "IMS CATALOG ENVIRONMENT REPORT"          PAGE: 1
5655-U08          DATE: 04/05/2021  TIME: 20.04.44          FABXCATM - V2.R2

IMS ENVIRONMENT

IMS ID          : SYS1
IMS VERSION     : 15.01.00

IMS CATALOG ENVIRONMENT

DFSDF MEMBER NAME      : DFSDFRGN
DFS3CDX0 ROUTINE      : NO
ACB MANAGEMENT        : ACBLIB

ALIAS OF CATALOG DB   : DFSC
CATALOG HLQ           : IMS.CATALOG
IMS DIRECTORY HLQ     : IMS.CATALOG

```

Figure 171. Example of the IMS Catalog Environment report

This report contains the following fields:

### IMS ID

IMS ID.

### IMS VERSION

IMS version. The utility, which can retrieve the IMS installed level of 15.2 or later, reports the IMS installed level instead of the IMS base level.

### 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 OF CATALOG DB

The alias name of the IMS catalog database.

### CATALOG HLQ

The high-level qualifier (HLQ) of the IMS catalog.

### IMS DIRECTORY HLQ

The high-level qualifier of the IMS directory data sets. This field is blank if the IMS directory was not referred to in the job.

## Error and Warning Messages report

This report contains error and warning messages. If error or warning messages were issued during the process, those messages are printed to the Error and Warning Messages report. If no error or warning messages were issued, this report contains "No message".

```
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "ERROR AND WARNING MESSAGES"          PAGE: 1
5655-U08          DATE: 04/05/2021  TIME: 20.04.44          FABXCATM - V2.R2

FABN0068W RDMVTAB CSECT IS CUSTOMIZED: MEMBER=DBD@CUST, TYPE=DBD TYPE ACB
FABN0077W VERSION PARAMETER DBD STATEMENT IS NOT DECODED. EXIT PARAMETERS OF DBD AND SEGM STATEMENTS ARE NOT DECODED
FABN0077W VERSION PARAMETER DBD STATEMENT IS NOT DECODED.
FABN0077W VERSION PARAMETER DBD STATEMENT IS NOT DECODED.
```

Figure 172. Example of the Error and Warning Messages report

## FABXCSRC data set (Convert function)

The FABXCSRC data set contains decoded IMS DBDGEN or IMS PSBGEN control statements.

```
      TITLE 'ASSEMBLE OF DBDNAME=DBD@D63A '
*      IMS DIRECTORY - ACTIVE
*      HLQ=IMSVS.IMS15A.DFSCD000
*      GENERATION DATE 04/01/2021 TIME 21.27.09
*      DECODE DATE 04/05/2021 TIME 20.04.44
*      IMS VERSION 15.1
      DBD      NAME=DBD@D63A,ACCESS=(HDAM,OSAM),RMNAME=(RNM,2,500,800),C
              PASSWD=NO,
              VERSION=,
              DATE 03/05/21 TIME 23.32
              ENCODING=CPDBD1,REMARKS='DBD_REMARKS_MAX256_123456789'
*****
*      DATASET GROUP NUMBER 1
*****
DSG001 DATASET DD1=DD@D63A,SIZE=(1690),SCAN=3,FRSPC=(2,3),SEARCHA=2,
      REMARKS='DBD@D03A_DD@D03A_REMARKS'
*****
*      SEGMENT NUMBER 1
*****
      SEGM      NAME=D03SEG1,PARENT=0,BYTES=100,RULES=(LLL, LAST),
              PTR=(TWIN,,,,)
      FIELD      NAME=(D03FLD1A,SEQ,U),START=1,BYTES=10,TYPE=C
*****
*      SEGMENT NUMBER 2
*****
      SEGM      NAME=D03SEG2,PARENT=((D03SEG1,)),BYTES=100,
              RULES=(LLL, LAST),PTR=(TWIN,,,,)
      FIELD      NAME=(D03FLD2A,SEQ,U),START=1,BYTES=10,TYPE=C
      DBDGEN
      FINISH
      END
```

Figure 173. Example of IMS DBDGEN control statements re-generated by the Catalog Manager utility

For restrictions that apply to generated control statements, see [“Catalog Manager utility restrictions”](#) on page 284.

## DBDSRC and PSBSRC data sets (Convert function)

The Catalog Manager utility creates IMS DBDGEN or PSBGEN control statements in the PDS or PDSE specified for the DBDSRC DD statement or the PSBSRC DD statement. These IMS DBDGEN or PSBGEN control statements are identical to those created in the FABXCSRC data set.

If the specified data set is not a PDS or PDSE, the program ends abnormally with an open error.

**Note:** If the member specified already exists in the PDS or PDSE, the program overrides the member.

## Output from the map function

Output from the Catalog Manager utility for generating maps and reports of the DBD or PSB consists of the FABXCRP0 data set, FABXCRP1 data set, and FABXCRP2 data set.

### FABXCRP0 data set (Map function)

The FABXCRP0 data set contains the Control Statement report, which shows the echo of the FABXCIN control statements and the selected runtime options.

```
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "CONTROL STATEMENT REPORT"          PAGE: 1
5655-U08          DATE: 04/01/2023  TIME: 03.03.30          FABXCATM - V2.R2

"CONTROL STATEMENTS"
0.....1.....2.....3.....4.....5.....6.....7.....8
1234567890123456789012345678901234567890123456789012345678901234567890

PROC  FUNC=MAP,
      INPUT=CATALOG_DB,
      INSTANCE=ACTIVE
REPORT MAP_REPORT=YES
DBD   NAME=DBD@D01A
END

"RUNTIME OPTIONS"
STATEMENT  KEYWORD          RUNTIME OPTIONS FOR THIS STEP
-----
PROC      FUNC              MAP

OPTIONAL STATEMENTS
STATEMENT  KEYWORD          RUNTIME OPTIONS FOR THIS STEP
-----
REPORT    MAP_REPORT        YES
```

Figure 174. Example of the Control Statement report

### FABXCRP1 data set (Map function)

The FABXCRP1 data set contains the FABXPPRM Echo Back report, the IMS Catalog Environment report, and, if errors or warning messages were issued, the Error and Warning Messages report.

Subsections:

- [“FABXPPRM Echo Back report” on page 324](#)
- [“IMS Catalog Environment report” on page 324](#)
- [“Error and Warning Messages report” on page 325](#)

### FABXPPRM Echo Back report

This report contains an echo of the FABXPPRM parameters.

```
IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "FABXPPRM ECHO BACK"          PAGE: 1
5655-U08          DATE: 04/01/2023  TIME: 03.03.30          FABXCATM - V2.R2

"FABXPPRM STATEMENT"
0.....1.....2.....3.....4.....5.....6.....7.....8
1234567890123456789012345678901234567890123456789012345678901234567890
DLI,FABXCAPL,DFSCP000,,,,,,,,,,,,,Y,N,,,,,,,,,,,,,DFSDF=RGN
```

Figure 175. Example of the FABXPPRM Echo Back report

### IMS Catalog Environment report

This report contains environment information about the IMS system and the IMS catalog.

```

IMS ENVIRONMENT
IMS ID           : SYS1
IMS VERSION      : 15.03.00

IMS CATALOG ENVIRONMENT
DFSDF MEMBER NAME : DFSDFRGN
DFS3CDX0 ROUTINE  : NO
ACB MANAGEMENT   : CATALOG

ALIAS OF CATALOG DB : DFSC
CATALOG HLQ       : IMS.CATALOG.DFSCD000
IMS DIRECTORY HLQ  : IMS.CATALOG.DFSCD000
    
```

Figure 176. Example of the IMS Catalog Environment report

This report contains the following fields:

**IMS ID**

IMS ID.

**IMS VERSION**

IMS version. The utility, which can retrieve the IMS installed level of 15.2 or later, reports the IMS installed level instead of the IMS base level.

**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 OF CATALOG DB**

The alias name of the IMS catalog database.

**CATALOG HLQ**

The high-level qualifier (HLQ) of the IMS catalog.

**IMS DIRECTORY HLQ**

The high-level qualifier of the IMS directory data sets. This field is blank if the IMS directory was not referred to in the job.

**Error and Warning Messages report**

This report contains error and warning messages. If error or warning messages were issued during the process, those messages are printed to the Error and Warning Messages report. If no error or warning messages were issued, this report contains "No message".

```

FABX0567W DBD PRPR01P IS NOT REFFERRED BY DBD PRGR01P.
FABX0567W DBD PRIP01P IS NOT REFFERRED BY DBD PRGR01P.
FABX0567W DBD PRST01P IS NOT REFFERRED BY DBD PRGR01P.
    
```

Figure 177. Example of the Error and Warning Messages report

## FABXCRP2 data set (Map function)

The FABXCRP2 data set contains the DBD map, DBD report, PSB map, PSB report, and PSB summary report.

### Referencing external DBDs

When the Catalog Manager utility processes a DBD or PSB which refers to an external DBD, the utility also processes the external DBD with the following rules because several DBD instances may be stored in the IMS catalog database and the utility should identify the DBD instance of external DBD to be processed.

The Catalog Manager utility processes an external DBD when processing of a DBD or PSB instance refers to the DBD and meets one of the following conditions:

- Active instance
- Pending instance
- The latest instance of DBD which has the same database versioning number
- The latest instance of LOGICAL DBD

When the Catalog Manager utility processes a Pending DBD or PSB instance, it refers to a Pending instance of external DBD. If the utility cannot find a Pending instance of external DBD, it refers to an Active instance of external DBD. When the utility processes a DBD or PSB other than Pending instance, the utility refers to an Active instance of external DBD. When the utility cannot find or does not refer to an external DBD instance, it generates maps without external DBD instance. When the utility cannot find or does not refer to an external DBD while processing a PSB, a PSB report is not generated.

Because the IMS catalog database may or may not have Active or Pending timestamp in its resource header segment, the Catalog Manager utility obtains Active or Pending timestamp in the following ways:

#### **IMS catalog database has timestamp of Active or Pending instance in its header segment**

The Catalog Manager utility obtains Active or Pending timestamp in the header segment of the IMS catalog database.

#### **IMS catalog database does not have timestamp of Active or Pending instance in its header segment**

When you create an IMS catalog database without `MANAGEDACBS=` statement of the DFS3PU00 utility (IMS catalog populate utility), no Active or Pending timestamp is stored in the header segment of the IMS catalog database.

Also, when you use the old IMS catalog database, it may not have a timestamp of Active or Pending instance. In this case, the Catalog Manager utility uses a timestamp of the IMS directory data set as Active timestamp and the IMS directory staging data set as Pending timestamp when `ACBMGMT=CATALOG` is specified in the DFSDFxxx PROCLIB member. When `ACBMGMT=ACBLIB` is specified in the DFSDFxxx PROCLIB member, the Catalog Manager utility uses the ACB library specified with ACBLIB DD to obtain an Active timestamp or ACBLIBS DD to obtain a Pending timestamp.

### ***DBD and PSB maps***

The Catalog Manager utility generates DBD and PSB maps.

The utility generates the maps as follows:

- A DBD map for each DBD instance in the IMS catalog database.
- A PSB map for each database PCB within the PSB instance in the IMS catalog database.

Subsections:

- [“Format of the maps” on page 327](#)
- [“Differences from DBD, PSB, and ACB maps generated by the DBD/PSB/ACB Mapper utility” on page 328](#)
- [“Report field description” on page 328](#)
- [“Example of the DBD map” on page 328](#)



- [“Example of the PSB map” on page 328](#)

## Format of the maps

The maps produced by the Catalog Manager utility depict the hierarchical structure of a database as described in a DBD. The DBD map heading shows the DBD member name, and ACTIVE or PENDING status. If the status is neither ACTIVE nor PENDING, \*\*\*\*\* is shown. The map also shows the database versioning number, its creation date, time, and IMS version of the DBD. It also shows the access method, or if it is a logical database LOGICAL, for the DBD currently being mapped.

The PSB map heading shows the PSB member name, and ACTIVE or PENDING status. If the status is neither ACTIVE nor PENDING, \*\*\*\*\* is shown. The map also shows its creation date, time, and IMS version of the PSB. PSB map also shows its referenced DBD.

When the utility does not refer to an external DBD, its status, DBVER=, gendate, and ACCESS= in the header DBDNAME= section are shown with asterisks (\*).

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 24. Data set group characters

Character	Explanation
*	The first data set group
+	The second data set group
"	The third data set group
.	The fourth data set group
=	The fifth data set group
-	The sixth data set group
#	The seventh data set group
%	The eighth data set group
;	The ninth data set group
'	The 10th data set group
V	Virtual logical child
C	Pointer and parent segment concatenation

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 9999 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 from DBD, PSB, and ACB maps generated by the DBD/PSB/ACB Mapper utility

The Catalog Manager utility generates maps from the stored values of the specified instance in the IMS catalog database. Although the format of the map is similar, certain elements are different between the maps generated by the Catalog Manager utility and those by the DBD/PSB/ACB Mapper utility.

### DBD maps

LOGICAL DBDs are stored in the IMS catalog database, so the Catalog Manager utility generates a map of the LOGICAL DBD.

Only one DBD of shared secondary index is stored in the IMS catalog database, so the utility generates a map for the one stored in the IMS catalog database.

When MULT for DEDB is not stored in the IMS catalog database, it is not shown in the BOX.

### PSB maps

Even when the utility cannot find a referenced DBD instance in the IMS catalog database, the utility generates a PSB map from the IMS catalog database. When the utility cannot find a referenced DBD instance in the IMS catalog database, it generates a map with only the database PCB values stored in the IMS catalog database. In this case, some descriptions such as segment code, VAR, and 2ND cannot be described in the box.

## Report field description

For details about report field description of DBD and PSB maps, see [“DBD, PSB, and ACB maps”](#) on page 210 of the DBD/PSB/ACB Mapper utility.

## Example of the DBD map

The following figure shows an example of the DBD map.

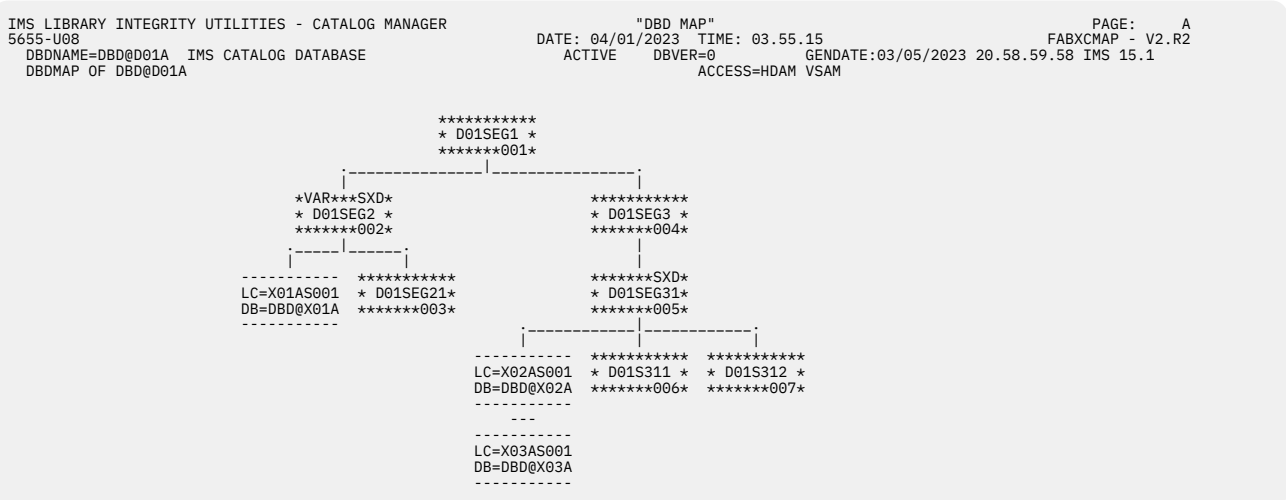


Figure 178. Example of the DBD map

## Example of the PSB map

The following figure shows an example of the PSB map.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER
5655-U08
PSBNAME=PSB@001 IMS CATALOG DATABASE
DBDNAME=DBD@H01A IMS CATALOG DATABASE
DBDMAP OF DBD@H01A IN PSB-PSB@001

"PSB MAP"
DATE: 04/01/2023 TIME: 04.04.09
ACTIVE
ACTIVE DBVER=0
ACCESS=HIDAM VSAM

PAGE: A
FABXCMAP - V2.R2
GENDATE:03/05/2023 20.58.59.58 IMS 15.1
GENDATE:03/05/2023 20.58.59.58 IMS 15.1

*VAR*****
* H01ASEG1*
*R*****001*
-----
*****
* H01ASEG2* * H01ASEG3* *VAR***** * H01ASEG5*
*G*****002* *R*****003* *G*****004* *G*****005*
*****
* H01ASG51*
*G*****006*

```

Figure 179. Example of the PSB map

## DBD and PSB reports

The Catalog Manager utility generates DBD and PSB reports from the IMS catalog database. Each report contains details about the DBD or PSB.

The utility generates the reports as follows:

- When processing a DBD, a DBD report is generated.
- When processing a PSB, a PSB report is generated.

The DBD and PSB reports show the following information:

- Database information
- Data set group or DEDB area information
- Segment information
- Field information

Subsections:

- [“Differences from DBD, PSB, and ACB reports generated by the DBD/PSB/ACB Mapper utility” on page 329](#)
- [“Report field description” on page 330](#)
- [“Example of the DBD report” on page 330](#)
- [“Example of the PSB report” on page 330](#)

## Differences from DBD, PSB, and ACB reports generated by the DBD/PSB/ACB Mapper utility

The Catalog Manager utility generates reports from the stored values of the specified instance in the IMS catalog database. Although the format of the report is similar, certain elements are different between the reports generated by the Catalog Manager utility and those by the DBD/PSB/ACB Mapper utility.

The Catalog Manager utility reports the values stored in the IMS catalog database, so the values that are not stored in the IMS catalog database are shown as N/A in the report.

Example of the N/A value:

- SEGMENT LENGTH MAX, MIN
- PFS LEN, PFX+MAX

The report generated by the Catalog Manager utility has the following differences:

- **\*\*SEARCH\*\***, **\*\*SUBSEQ\*\***, and **\*\*SOURCE\*\*** show only field name. Bytes and start position are omitted.
- The type of pointers in the PSB report is calculated by using DBD, so the pointer is the same as DBD in the DBD report.
- The values such as RULES, EXIT, or LCHILD PTR= are reported with the values stored in the IMS catalog database.

- \*PR\* is shown only when the utility obtains the information from the DBD being processed.

## Report field description

For details about report field description of DBD and PSB reports, see [“DBD, PSB, and ACB reports” on page 216 of the DBD/PSB/ACB Mapper utility.](#)

## Example of the DBD report

The following figure shows an example of the DBD report.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "DBD REPORT"          PAGE: 1
5655-U08          DATE: 04/01/2023 TIME: 10.44.51          FABXCMAP - V2.R2
DBDNAME=DBD@D10A IMS CATALOG DATABASE          ACTIVE DBVER=1          GENDATE:03/20/2023 10.29.54.17 IMS 15.1
DBDMAP OF DBD@D10A          ACCESS=HDAM VSAM

          DDDD BBBB DDDD @@@ DDDD 1 000 AAA
          D D B B D D @ @ D D 11 0 0 A A
          D D B B D D @ @ D D 1 0 00 A A
          D D BBBB D D @ @ D D 1 0 0 0 AAAAA
          D D B B D D @ @ D D 1 00 0 A A
          D D B B D D @ @ D D 1 0 0 A A
          DDDD BBBB DDDD @@@@ DDDD 11111 000 A A

VERSION=03/20/23 10.29

LEVELS= 1 SEGMENTS= 1 DATA SET GROUPS= 1
RANDOMIZING ROUTINE=DFSHOC10 ROOT ANCHOR POINTS= 2 MAX RBN= 23 BYTES= 2048
ID= 1* DD@D03A PRIME DS LOG. RCD. LEN= 0, BLOCKSIZE= 0 SEGMENT LENGTH MAX= N/A, MIN= N/A
OFLW DS LOG. RCD. LEN= 0, BLOCKSIZE= 0 KEY LENGTH MAX= N/A, MIN= N/A NUMBSEG= 1
SEG-NAME SC# LV PAR -LEN- ---FREQ--- C P L L P P P L L E RULES PHYS. N-SEQ OR INSRT LOG CHLD
R FB FB .F .L .F .L S I D R INSRT FLD-NAME LEN STRT PNTR RULES
D03SROOT* 1 1 0 32 0.00 XX FB FB .F .L .F .L S I D R L L L LAST
          *PFX* LEN= N/A MAX= 32 PFX+MAX= N/A
          *FLD* D03FLDA 8 1 CHARACTER SEQ, UNIQUE
          *FLD* D03FLDB 12 9 HEXADECIMAL
  
```

Figure 180. Example of the DBD report

## Example of the PSB report

The following figures show an example of the PSB report.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "PSB REPORT"          PAGE: 1
5655-U08          DATE: 04/01/2023 TIME: 04.04.09          FABXCMAP - V2.R2
PSBNAME=PSB@001 IMS CATALOG DATABASE          ACTIVE DBVER=0          GENDATE:03/05/2023 20.58.59.58 IMS 15.1
DBDNAME=DBD@D01A IMS CATALOG DATABASE          ACTIVE          GENDATE:03/05/2023 20.58.59.58 IMS 15.1
DBDMAP OF DBD@D01A IN PSB-PSB@001          ACCESS=HDAM VSAM

          PPPP SSS BBBB @@@ 000 000 1
          P P S S B B @ @ 0 0 0 0 11
          P P S S B B @ @ 0 0 0 0 00 1
          PPPP SSS BBBB @ @ 0 0 0 0 0 0 1
          P S S B B @ @ @ 0 0 0 0 0 1
          P S S B B @ @ 0 0 0 0 1
          P SSS BBBB @@@@ 000 000 11111

          DDDD BBBB DDDD @@@ DDDD 000 1 AAA
          D D B B D D @ @ D D 0 0 11 A A
          D D B B D D @ @ D D 0 0 1 A A
          D D BBBB D D @ @ D D 0 0 0 1 AAAAA
          D D B B D D @ @ D D 0 0 0 1 A A
          D D B B D D @ @ D D 0 0 0 1 A A
          DDDD BBBB DDDD @@@@ DDDD 000 11111 A A

LEVELS= 4 SEGMENTS= 7 DATA SET GROUPS= 1
RANDOMIZING ROUTINE=RNM ROOT ANCHOR POINTS= 2 MAX RBN= 500 BYTES= 800
ID= 1* DD@D01A CHANGED DATA EXIT=* ,KEY ,DATA ,NOPATH, (CASCADE,KEY ,LOG
PRIME DS LOG. RCD. LEN= 0, BLOCKSIZE= 0 SEGMENT LENGTH MAX= N/A, MIN= N/A
OFLW DS LOG. RCD. LEN= 0, BLOCKSIZE= 0 KEY LENGTH MAX= N/A, MIN= N/A NUMBSEG= 7
FREE BUFFER PCT= 2, RECORD PCT= 3
SEG-NAME SC# LV PAR -LEN- ---FREQ--- C P L L P P P L L E RULES PHYS. N-SEQ OR INSRT LOG CHLD
R FB FB .F .L .F .L S I D R INSRT FLD-NAME LEN STRT PNTR RULES
D01SEG1 * 1 1 0 100 0.00 XX FB FB .F .L .F .L S I D R L L L LAST
          CMPRS RTNE-A INIT-EP
          CHANGED DATA EXIT-SAME AS DBD
          *PFX* LEN= N/A MAX= 100 PFX+MAX= N/A
          *FLD* D01FLD1A 10 1 CHARACTER SEQ, UNIQUE
          *FLD* D01FLD1B 10 11 HEXADECIMAL
          VAR LEN
          KEY-EP INIT-EP
D01SEG2 * 2 2 1 96 0.00 X XX L L L LAST SNDIXD
          CMPRS RTNE-A
          CHANGED DATA EXIT
          EXIT=EXITA ,KEY ,DATA ,NOPATH
          (CASCADE,KEY ,DATA ,NOPATH)
          NOLOG
          *PFX* LEN= N/A MAX= 96 PFX+MAX= N/A
          MIN= 64 PFX+MIN= N/A
          *FLD* D01FLD2A 10 1 PACKED SEQ, UNIQUE
          *FLD* D01FLD2B 10 11 CHARACTER
          *XFD* XDF01D01 SECONDARY INDEXED FIELD
          **SRCSEG** D01SEG21
          CONSTANT=X'F1'
          *NULLEXIT* NULL=X'F0' EXIT-EX=XDF01
          **SEARCH** D01FD21A
          **SUBSEQ** /SX002
          **SOURCE** D01FD21B
  
```

Figure 181. Example of the PSB report (Part 1 of 2)

SEG-NAME	SC#	LV	PAR	-LEN-	---	FREQ---	C	P	P	L	L	P	P	L	L	E	RULES	PHYS.	SEG-NAME	D-B-NAME	FORM	LOG	CHLD
							T	T	P	T	P	H	C	C	C	C	P	I	D	R	OR	OR	INSRT
							R	FB	FB	FB	FB	.F	.L	.F	.L	S	I	D	R	OR	OR	PNTR	RULES
D01SEG21*	3	3	2	96		0.00	XX										L L L	FIRST	*FSRCSEG** D01SEG21				
																							FIX LEN
																							CHANGED DATA EXIT-SAME AS DBD
																							*PFX* LEN= N/A MAX= 96 PFX+MAX= N/A
																							*FLD* D01FD21A 10 1 HEXADECIMAL
																							*FLD* /SX002 4 1 SYSTEM-RELATED FIELD
																							*FLD* D01FD21B 3 16 CHARACTER
																							FIX LEN
D01SEG3 *	4	2	1	96		0.00	X X										L L L	LAST					CHANGED DATA EXIT
																							EXIT=EXITB ,NOKEY,DATA ,PATH
																							(CASCADE,NOKEY,NODATA,PATH )
																							LOG
																							*PFX* LEN= N/A MAX= 96 PFX+MAX= N/A
																							*FLD* D01FLD3A 10 1 CHARACTER
																							SEQ,MULTIPLE
																							FIX LEN
																							CHANGED DATA EXIT-SAME AS DBD
																							*PFX* LEN= N/A MAX= 96 PFX+MAX= N/A
																							*FLD* D01FL31A 10 1 CHARACTER
																							SEQ,UNIQUE
																							*XFD* XDF02D01 SECONDARY INDEXED FIELD
																							**SRCSEG** D01S311
																							**SEARCH** D01F311A
																							*FSRCSEG** D01S311
																							*XFD* XDF03D01 SECONDARY INDEXED FIELD
																							**SRCSEG** D01S312
																							**SEARCH** D01F312A
																							*FSRCSEG** D01S312
D01S311 *	6	4	5	96		0.00	X										L L L	HERE					FIX LEN
																							CHANGED DATA EXIT-SAME AS DBD
																							*PFX* LEN= N/A MAX= 96 PFX+MAX= N/A
																							*FLD* D01F311A 10 1 CHARACTER
																							*FLD* D01F311B 10 11 CHARACTER
																							FIX LEN
																							CHANGED DATA EXIT-SAME AS DBD
																							*PFX* LEN= N/A MAX= 96 PFX+MAX= N/A
																							*FLD* D01F312A 10 1 CHARACTER

Figure 182. Example of the PSB report (Part 2 of 2)

### PSB summary report

The PSB summary report provides overall information about the PSB, its attributes, and the PCBs it contains. 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
- Status (ACTIVE, PENDING, or \*\*\*\*\* if the status is neither ACTIVE nor PENDING)
- Date, time, and IMS version when the PSB was generated

Subsections:

- [“Differences from PSB summary report generated by the DBD/PSB/ACB Mapper utility” on page 331](#)
- [“Report field description” on page 332](#)
- [“Example of the PSB summary report” on page 332](#)

### Differences from PSB summary report generated by the DBD/PSB/ACB Mapper utility

The Catalog Manager utility generates a PSB summary report only with the PSB being processed. Therefore, the following differences exist between the PSB summary report generated by the Catalog Manager utility and the one by the DBD/PSB/ACB Mapper utility.

- The information about the database type is not reported because the Catalog Manager utility only uses the values of the PSB instance.
  - DL/I DATABASE ACCESS INDICATOR
  - FAST PATH DATABASE ACCESS
  - FAST PATH DATABASE UPDATE INTENT
  - FAST PATH UTILITY PROGRAM INDICATOR
  - MSDB PCB INDICATOR

- DEDB PCB INDICATOR
- The following value is not reported because the PSB summary report contains the same information.
  - NUMBER OF SENSITIVE SEGMENTS
- The following value is not reported because this value is not stored in the IMS catalog database.
  - DBLEVEL

## Report field description

For details about report field description of PSB summary report, see [“PSB and ACB summary reports”](#) on page 214 of the DBD/PSB/ACB Mapper utility.

## Example of the PSB summary report

The following figures show an example of the PSB summary report.

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "PSB SUMMARY REPORT"          PAGE: 1
5655-U08          DATE: 04/01/2023  TIME: 09.54.38          FABXCMAP - V2.R2
PSBNAME=PSB@003  IMS CATALOG DATABASE          ACTIVE          GENDATE:03/05/2023 20.58.59.58 IMS 15.1
PSB REPORT OF PSB@003

PPPPPPPP  SSSS  BBBB BBBB  @@@@@@  0000  0000  333333333333
PPPPPPPPPP  SSSSSSSS  BBBB BBBB  @@@@@@  00000000 0  00000000 0  333333333333
PP  PPP  SSS  SSS  BB  BBB  @@@  @@@  000  0000  000  0000  33  333
PP  PP  SS  SS  SS  BB  BB  @@  @@  00  00  00  00  00  333
PP  PP  SS  SS  BB  BB  @@  @@  00  000  00  000  00  333
PP  PP  SS  SS  BB  BB  @@  @@  00  0000  00  0000  00  333
PP  PPP  SSS  BBBB BBBB  @@  @@@@@@  00  00  00  00  00  00  33333333
PPPPPPPPPP  SSSSSS  BBBB BBBB  @@  @@  @@  00  00  00  00  00  00  3333333333
PPPPPPPP  SSSSS  BBBB BBBB  @@  @@  @@  00  00  00  00  00  00  333333333333
PP  SSS  SS  BB  BB  @@  @@  0000  00  0000  00  333
PP  SS  SS  BB  BB  @@  @@@@@@  000  00  000  00  33
PP  SS  SS  BB  BB  @@  @@@@@@  00  00  00  00  33  33
PP  SSS  SSS  BB  BBB  @@@  0000  000  0000  000  333  333
PPPP  SSSSSSSS  BBBB BBBB  @@@@@@  0 00000000 0 00000000 0 3333333333
PPPP  SSSS  BBBB BBBB  @@@@@@  0000  0000  333333

PSB PREFIX SUMMARY
600 BYTES NEEDED FOR I/O WORK AREA
840 BYTES NEEDED FOR SSA WORK AREA
100 I/O ERROR OPTION RETURN CODE
WTOR ON DATABASE I/O ERRORS
ON-LINE IMAGE COPY
MAX OF DATABASE Q COMMAND CALL = 10
PL/1 LANGUAGE

TP PCB SUMMARY
SYSTEM ALWAYS ADDS AN I/O PCB
NO USER TP PCB.

DB PCB SUMMARY
DB PCB # 1 DBD@S02A
DB PCB # 2 DBD@M02A
DB PCB # 3 DBD@S03A

GSAM PCB SUMMARY
NO GSAM PCBS

TP PCB DETAIL
SYSTEM ALWAYS ADDS AN I/O PCB
NO USER TP PCB.

DB PCB DETAIL

```

Figure 183. Example of the PSB summary report (Part 1 of 2)

```

IMS LIBRARY INTEGRITY UTILITIES - CATALOG MANAGER          "PSB SUMMARY REPORT"          PAGE: 2
5655-U08          DATE: 04/01/2023  TIME: 09.54.38          FABXCMAP - V2.R2
PSBNAME=PSB@003  IMS CATALOG DATABASE          ACTIVE          GENDATE:03/05/2023 20.58.59.58 IMS 15.1
PSB REPORT OF PSB@003

DB PCB # 1 DBD@S02A
PROCESSING OPTION = G
SEQUENTIAL BUFFERING = ACTIVATED CONDITIONALLY
SINGLE POSITIONING REQUESTED
LENGTH OF KEY FEEDBACK AREA = 30 (00001E HEX)
1 S02SEG1 **ROOT** G
2 S02SEG2 S02SEG1 G
3 S02SEG21 S02SEG2 G

DB PCB # 2 DBD@M02A
PROCESSING OPTION = G
SEQUENTIAL BUFFERING = NOT BE USED
SINGLE POSITIONING REQUESTED
LENGTH OF KEY FEEDBACK AREA = 30 (00001E HEX)
1 M02SEG1 **ROOT** G

DB PCB # 3 DBD@S03A
PROCESSING OPTION = G
SEQUENTIAL BUFFERING = NOT BE USED
SINGLE POSITIONING REQUESTED
LENGTH OF KEY FEEDBACK AREA = 30 (00001E HEX)
1 S03SEG1 **ROOT** G

GSAM PCB DETAIL
NO GSAM PCBS

```

Figure 184. Example of the PSB summary report (Part 2 of 2)

---

# Chapter 11. 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 333](#)
- [“Generating application control blocks” on page 334](#)
- [“Merging Advanced ACBGEN load modules into the IMS SDFSRESL library” on page 334](#)
- [“Using the Advanced ACBGEN utility in an ACB Generation and Catalog Populate utility job” on page 335](#)
- [“JCL requirements for the Advanced ACBGEN utility” on page 336](#)
- [“Control statements for the Advanced ACBGEN utility” on page 338](#)
- [“Output from the Advanced ACBGEN utility” on page 342](#)

---

## 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*, DFSDLBLO. 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 DFSDLBLO 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, DFSDLBLO, 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 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 336.

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 338.

3. Submit the job.

4. Check the output data sets that are generated.

See [“Output from the Advanced ACBGEN utility”](#) on page 342.

## 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 334) 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 336.

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 338.

3. Submit the job.

4. Check the output data sets that are generated.

See [“Output from the Advanced ACBGEN utility”](#) on page 342.

## 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” on page 336](#)
- [“Common DD statements that are used differently” on page 336](#)
- [“Common DD statements” on page 337](#)
- [“DD statements used in ACB Generation and Catalog Populate utility \(DFS3UACB\) jobs” on page 338](#)
- [“Unique DD statements for the Advanced ACBGEN utility” on page 338](#)

### 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=DFSRR00
```

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*.

**COMPCTL 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 339](#).

### 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” on page 339](#)
- [“Syntax rules” on page 339](#)
- [“ACBGEN command” on page 341](#)

### 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 185. 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 ignored. 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 *prevue* 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 a *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**

The Input Specifications 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” on page 342](#)
- [“Report field descriptions” on page 343](#)

#### **Sample report**

The following figure shows an example of the Input Specifications report.



+-----+  
 |INPUT SPECIFICATIONS|  
 +-----+

CONTENTS OF "ACBSYSIN" CONTROL STATEMENT DATASET:  
 =====

----	+	----	1----	+	----	2----	+	----	3----	+	----	4----	+	----	5----	+	----	6----	+	----	7--	RCD#
			ACBGEN			REPORTS=(SYSINLST=YES,PSBLIST=YES)																0001

CONTENTS OF "SYSIN " CONTROL STATEMENT DATASET:  
 =====

----	+	----	1----	+	----	2----	+	----	3----	+	----	4----	+	----	5----	+	----	6----	+	----	7--	RCD#
			BUILD			PSB=(PSB@001,PSB@002)																000001
			BUILD			DBD=(DBD@M04A)																000002
			DELETE			DBD=(DBD@M03A)																000003

CONTENTS OF EXEC STATEMENT PARM FIELD:  
 =====

UPB

RUN-TIME PARAMETERS:  
 =====

```

IMS RELEASE LEVEL BEING USED.....15.1.0
TIMESTAMP USED.....21.274 16:19:45
BUILD PSB=ALL SPECIFIED.....N
PRE-COMPRESSION.....N
POST-COMPRESSION.....N
OPTIONAL REPORTS REQUESTED:
  SYSIN BUILD AND DELETE LISTS.....Y
  FINAL PSB BUILD LIST.....Y
  LOAD MODULE MGMT STATS.....Y
  DFSPRINT FOR DFS-TYPE MESSAGES.....N
PROGRESS MONITOR REQUESTED.....N
  
```

Figure 186. 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, or 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.

**Note:** Even when the IMS installed level is updated to 15.2, IMS 15.1 is still used for several resources such as ACB. Advanced ACBGEN reports the same IMS version as the IMS ACBGEN utility.

### 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

The ACB/PSB/DBD Library Information 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.

```
IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE: 2
5655-U08                DATE: 10/01/2021  TIME: 15.50.12        FABQMAIN - V2.R2

+-----+
|ACB/PSB/DBD LIBRARY INFORMATION|
+-----+

DDNAME  DSNAME                NUMBER  DIR BLOCKS
-----  -----                -
MEMBRS  ALLOC   USED
-----  -----
IMSACB  IMSVS.ACBLIB            5        10      1
IMS     IMSVS.PSBLIB           42       40      7
        IMSVS.DBDLIB           47       40      7
```

Figure 187. ACB/PSB/DBD Library Information report

## DBD/PSB Names Specified via SYSIN report

The DBD/PSB Names Specified via SYSIN 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.

```
IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE: 3
5655-U08                DATE: 10/01/2021  TIME: 16.19.45        FABQMAIN - V2.R2

+-----+
|DBD/PSB NAMES SPECIFIED VIA SYSIN|
+-----+

      BUILD LIST CONTENTS
      =====

      NAME      TYPE
      -----
      DBD@M04A  DBD
      PSB@001   PSB
      PSB@002   PSB

      DELETE LIST CONTENTS
      =====

      NAME      TYPE
      -----
      DBD@M03A  DBD
```

Figure 188. DBD/PSB Names Specified via SYSIN report

## ACB Library Members Deleted Due to User Request report

The ACB Library Members Deleted Due to User Request 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=(SYSINLST=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.

```
IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE: 4
5655-U08                DATE: 10/01/2021  TIME: 16.19.45        FABQMAIN - V2.R2

+-----+
|ACB LIBRARY MEMBERS DELETED DUE TO USER REQUEST|
+-----+

DBD@M03A *PSB@006
NOTE: AN ASTERISK (*) INDICATES NAME WAS INCLUDED IMPLICITLY
```

Figure 189. ACB Library Members Deleted Due to User Request report

## Final PSB Build List report

The Final PSB Build List 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=(*dbdname*)

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.

```

IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE: 5
5655-U08                DATE: 10/01/2021  TIME: 16.19.45        FABQMAIN - V2.R2

+-----+
|FINAL PSB BUILD LIST|
+-----+

PSB@001  PSB@002  *PSB@007

          NUMBER OF PSB = 3          NUMBER OF DBD = 1
          NOTE: AN ASTERISK (*) INDICATES NAME WAS INCLUDED IMPLICITLY

```

Figure 190. Final PSB Build List report

## PSB Size Summary report

The PSB Size Summary 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” on page 346](#)
- [“Report field descriptions” on page 346](#)

## Sample report

The following figure contains an example of the PSB Size Summary report.

```

IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE: 7
5655-U08                DATE: 10/01/2021  TIME: 16.19.45        FABQMAIN - V2.R2

+-----+
|PSB SIZE SUMMARY|
+-----+

PSBNAME      TOTAL   PSB   PCB   WORK  <-----WORKAREA BREAKOUT----->   CSA   SAS
              SIZE    SIZE  SIZE AREA  NDX   XIO   IOA   SEG   SSA   SIZE  SIZE
-----
PSB@001      14528  12416  3040  2112   448   272  1000  152  200  2880  9280
PSB@002       7616   6016  1032  1600   264   272   600  152  280  1152  4736
PSB@007      1344    640   464   704    56     8   600    8    0   576    0

      MAX:    14528  12416  3040  2112   448   272  1000  152  280  2880  9280
      AVG:                   1536  4672

NOTE: THIS REPORT CONTAINS ALL OF THE DATA FOUND IN THE FOLLOWING DFS MESSAGES:
      0589, 0591, 0593, 0940, 0941, AND 0942.

```

Figure 191. 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**

The PSB/DBD Change Summary report consolidates all the information provided in the DFS0940 and DFS0960 messages into one print line per DBD or PSB, plus provides some information about those DBDs and PSBs. All DBDs are listed first followed by all PSBs in the ascending name sequence.

Subsections:

- [“Sample report” on page 347](#)
- [“Report field descriptions” on page 348](#)

**Sample report**

The following figure shows an example of the PSB/DBD Change Summary report.

+-----+  
|PSB/DBD CHANGE SUMMARY|  
+-----+

DBD/PSB NAME	TYPE	ACTION TAKEN	OLD SIZE	NEW SIZE	OLD GEN DATE	NEW GEN DATE	#DMBS REFERENCED
PSB@001	PSB	ADDED		12416		10/01/21	15
PSB@002	PSB	ADDED		6016		10/01/21	7
PSB@006	PSB	DELETED	512		10/01/21		
PSB@007	PSB	REPLACED	640	640	10/01/21	10/01/21	1

DBD/PSB NAME	TYPE	ACTION TAKEN	OLD SIZE	NEW SIZE	OLD GEN DATE	NEW GEN DATE
DBD@D01A	DBD	ADDED		704		10/01/21
DBD@D03A	DBD	ADDED		1280		10/01/21
DBD@E01A	DBD	ADDED		1408		10/01/21
DBD@E02A	DBD	ADDED		2560		10/01/21
DBD@H01A	DBD	ADDED		640		10/01/21
DBD@H02A	DBD	ADDED		1472		10/01/21
DBD@ISAM	DBD	ADDED		640		10/01/21
DBD@I01A	DBD	ADDED		768		10/01/21
DBD@I02A	DBD	ADDED		448		10/01/21
DBD@I03A	DBD	ADDED		960		10/01/21
DBD@M02A	DBD	ADDED		256		10/01/21
DBD@M03A	DBD	DELETED	192		10/01/21	
DBD@M04A	DBD	REPLACED	192	192	10/01/21	10/01/21
DBD@S02A	DBD	ADDED		704		10/01/21
DBD@X01A	DBD	ADDED		640		10/01/21
DBD@X02A	DBD	ADDED		448		10/01/21
DBD@X03A	DBD	ADDED		448		10/01/21
DBD@X04A	DBD	ADDED		448		10/01/21
DBD@X05A	DBD	ADDED		448		10/01/21
DBD@X06A	DBD	ADDED		640		10/01/21
DBD@X07A	DBD	ADDED		448		10/01/21
DBD@X08A	DBD	ADDED		448		10/01/21

NOTE: THIS REPORT REFLECTS THE ACTION TAKEN IN THE FOLLOWING DFS MESSAGES:  
0587, 0938, 0940, AND 0960.

Figure 192. 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

The Miscellaneous DFS Messages 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.

```
IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE: 9
5655-U08                DATE: 10/01/2021  TIME: 10.57.09        FABQMAIN - V2.R2

+-----+
|MISCELLANEOUS DFS MESSAGES|
+-----+

NOTE: THIS REPORT INCLUDES ALL DFS MESSAGES EXCEPT FOR:
      0589, 0591, 0593, 0649, 0938, 0940, 0941, 0942, AND 0960

MESSAGE MESSAGE
ID        TEXT
-----
DFS0586I DBD 'DTA1      ' REQUESTED IN DELETE OPERATION NOT FOUND IN ACBLIB - REQ
DFS0586I DBD 'DI21XXXX' REQUESTED IN BUILD  OPERATION NOT FOUND IN ACBLIB - REQ
DFS0586I DBD 'DTA1      ' REQUESTED IN BUILD  OPERATION NOT FOUND IN ACBLIB - REQ
DFS0929I BLDL FAILED FOR MEMBER --PSB1
DFS0587I ERROR BUILDING PSB=PSB1      - IT WILL BE DELETED FROM ACBLIB.  *****
DFS0929I BLDL FAILED FOR MEMBER --PSB2
DFS0587I ERROR BUILDING PSB=PSB2      - IT WILL BE DELETED FROM ACBLIB.  *****
DFS0929I BLDL FAILED FOR MEMBER --PSB3
DFS0587I ERROR BUILDING PSB=PSB3      - IT WILL BE DELETED FROM ACBLIB.  *****
DFS0962I DBD DI21XXXX NOT PROCESSED. NO ACBLIB PSB REFERENCES THE NAMED DBD.
DFS0590I END OF ACBLIB MAINTENANCE. HIGHEST CONDITION CODE WAS 00000016
```

Figure 193. Miscellaneous DFS Messages report

## DFS Messages Summary report

The DFS Messages Summary 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.

```

IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE: 10
5655-U08                DATE: 10/01/2021  TIME: 15.50.12        FABQMAIN - V2.R2

                                +-----+
                                |DFS MESSAGES SUMMARY|
                                +-----+

MESSAGE ID      QTY      MESSAGE TEXT (PARAPHRASED)
-----
DFS0589         3  PROCESSING COMPLETED FOR PSB
DFS0590         1  END OF ACBLIB MAINTENANCE
DFS0591         1  MAX SIZES OF PSB COMPONENTS
DFS0593         3  PSB WORKAREA BREAKOUT
DFS0938         1  DELETE PROCESSING COMPLETED FOR DBD
DFS0938         1  DELETE PROCESSING COMPLETED FOR PSB
DFS0940        21  DBD HAS BEEN ADDED/REPLACED IN ACBLIB
DFS0940         3  PSB HAS BEEN ADDED/REPLACED IN ACBLIB
DFS0941         3  PSB CSA AND SAS SIZES IF USING DL/I SEPARATE ADDRESS SPACE
DFS0942         1  MAX AND AVERAGE CSA AND SAS SIZES IF USING SEPARATE ADDR SPACE

```

Figure 194. DFS Messages Summary report

## Run Summary report

The Run Summary 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” on page 350](#)
- [“Report field descriptions” on page 351](#)

## Sample report

The following figure shows an example of the Run Summary report.



```

+-----+
|RUN SUMMARY|
+-----+

NUMBER OF BUILD PSB FAILURES..... 0
NUMBER OF PSBS ADDED TO ACBLIB..... 2
NUMBER OF PSBS DELETED FROM ACBLIB..... 0
NUMBER OF PSBS REPLACED IN ACBLIB..... 0

NUMBER OF DMBS ADDED TO ACBLIB..... 20
NUMBER OF DMBS DELETED FROM ACBLIB..... 1
NUMBER OF DMBS REPLACED IN ACBLIB..... 1
NUMBER OF DMBS NOT REPLACED IN ACBLIB..... 0

NUMBER OF LOAD PSBS ISSUED..... 3
NUMBER OF LOAD DBDS ISSUED..... 25

NUMBER OF ACBLIB MEMBERS..... 25
NUMBER OF ACBLIB DIRECTORY BLKS USED..... 5
NUMBER OF ACBLIB DIRECTORY BLKS UNUSED..... 5

START TIME.....15:50:12
END TIME.....15:50:45

COMPLETION CODE..... 0000
  
```

Figure 195. Run Summary report

## 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” on page 352](#)
- [“Report field descriptions” on page 353](#)

**Sample report**

The following figure shows an example of the Load Module Management Status report.

```
+-----+
|LOAD MODULE MGMT STATS|
+-----+

MAX ENTRYS 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 ENTRYS 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 196. 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. DMB SIZE = 00000576 BYTES
DFS0943I PSB PSBLIU01 REQUIRES MIN OF 00000000 AND MAX OF 00000028 BYTES OF STORAGE IN EPCB POOL IF USING FAST PATH.
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

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 197. 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.

```

14.48.44 JOB07119 +FABQ9997I 0005000 PSBS TO BE PROCESSED BY ACBGEN
15.04.01 JOB07119 +FABQ9998I 0001000 OF 0005000 PSBS PROCESSED BY ACBGEN
15.04.55 JOB07119 +FABQ9998I 0002000 OF 0005000 PSBS PROCESSED BY ACBGEN
15.05.52 JOB07119 +FABQ9998I 0003000 OF 0005000 PSBS PROCESSED BY ACBGEN
15.06.50 JOB07119 +FABQ9998I 0004000 OF 0005000 PSBS PROCESSED BY ACBGEN
15.07.50 JOB07119 +FABQ9998I 0005000 OF 0005000 PSBS PROCESSED BY ACBGEN
15.07.51 JOB07119 +FABQ9999I 0005000 PSBS PROCESSED BY ACBGEN

```

Figure 198. Messages in the MVS console and the JES job listing

---

## Chapter 12. 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 355](#)
- [“Analyzing ACB libraries” on page 355](#)
- [“JCL requirements for the ACBLIB Analyzer utility” on page 355](#)
- [“ACBSYSIN control statements” on page 357](#)
- [“Output from the ACBLIB Analyzer utility” on page 358](#)

---

### 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 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 355](#).
2. Optionally, code the control statements for ACBLIB Analyzer in the ACBSYSIN data set.  
See [“ACBSYSIN control statements” on page 357](#).
3. Submit the job.
4. Check the output data sets that are generated.  
See [“Output from the ACBLIB Analyzer utility” on page 358](#).

---

### 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” on page 356](#)
- [“EXEC statement” on page 356](#)
- [“DD statements” on page 356](#)

## JCL examples

The following JCL example reports on an ACB library.

```
//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 199. 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.

```
//ANALYZE EXEC PGM=FABQCHEK
//STEPLIB DD DSN=HPS.SHPSLMD0,DISP=SHR
//SYSPRINT DD SYSOUT=*
//DBTSNAP DD SYSOUT=*
//IMSDDB DD DISP=SHR,DSN=IMSVS.DBDLIB
//ACBSYSIN DD *
LISTLIB LIBTYPE=DBD,INDD=IMSDDB
/*
```

Figure 200. 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 358.

### 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 357](#).

## 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” on page 357](#)
- [“Syntax rules” on page 357](#)
- [“LISTLIB command” on page 357](#)

### Control statement example

The following figures show examples of ACBSYSIN control statements.

```
LISTLIB  
LISTLIB LIBTYPE=ACB,SNAP=(DIRENTRY=(Y,100),DIRRCD=N)  
LISTLIB LIBTYPE=ACB,USESORT=YES  
LISTLIB LIBTYPE=ACB,GENDATE=YES
```

*Figure 201. 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 202. 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 339](#) 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.

**USESORT=**

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

The Input Specifications report contains the information that was specified as input to this execution of the ACBLIB Analyzer utility.

Subsections:

- [“Sample report” on page 359](#)
- [“Report field descriptions” on page 359](#)

### Sample report

The following figure shows an example of the Input Specifications report.

```
IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE: 1
5655-U08                DATE: 10/01/2021  TIME: 16.19.45        FABQCHEK - V2.R2

                        +-----+
                        |INPUT SPECIFICATIONS|
                        +-----+

CONTENTS OF "ACBSYSIN" CONTROL STATEMENT DATASET:
=====
-----1-----2-----3-----4-----5-----6-----7--   RCD#
LISTLIB LIBTYPE=ACB ,SNAP=(DIRENTRY=N,DIRRCD=N)                 0001
```

Figure 203. Input Specifications report (ACBLIB Analyzer utility)

### Report field descriptions

#### CONTENTS OF "ACBSYSIN" CONTROL STATEMENT DATESET

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

The Library Information 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” on page 360](#)
- [“Report field descriptions” on page 360](#)

## Sample report

The following figure shows an example of the Library Information report.

```

IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE:      2
5655-U08                  DATE: 10/01/2021  TIME: 17.07.07      FABQCHECK - V2.R2

      +-----+
      |LIBRARY INFORMATION|
      +-----+

              LIBRARIES USED
              =====

DDNAME      LIB  DSNAME                                VOLSER  HIGH-TTR
-----
IMSACB      ACB  DBT005.LARGE.ACBLIB                  DBT005  '000113'

              DIRECTORY INFORMATION
              =====

DDNAME      NUMBER  DIR BLOCKS      BLOCK  DIRECTORY  DIRECTORY
-----      MEMBRS  ALLOC  USED      SIZE  ENTRY SIZE  ENTRYS/BLK
IMSACB              25          10      5    6233      40          6

```

Figure 204. Library Information report

## Report field descriptions

### 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

The Library Contents 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 360](#)
- [“Report field descriptions” on page 362](#)

## Sample report

The following figure shows an example of the Library Contents report when GENDATE=YES is not specified in the LISTLIB command.

+-----+  
 |LIBRARY CONTENTS|  
 +-----+

DMB DETAIL  
 =====

MBRNAME	TYPE	IMS LVL	SIZE (BYTES)	# DMBS REFERENCE	# PSBS REFERENCNG	COMMENTS
DBD@D01A	DMB	15.1	648		1	
DBD@D03A	DMB	15.1	1,192		1	
DBD@E01A	DMB	15.1	1,336		1	
DBD@E02A	DMB	15.1	2,504		1	
DBD@H01A	DMB	15.1	600		1	
DBD@H02A	DMB	15.1	1,344		2	
DBD@ISAM	DMB	15.1	560		1	
DBD@I01A	DMB	15.1	680		1	
DBD@I02A	DMB	15.1	392		1	
DBD@I03A	DMB	15.1	872		1	
DBD@M02A	DMB	15.1	200		1	
DBD@M04A	DMB	15.1	120		1	
DBD@S02A	DMB	15.1	616		1	
DBD@X01A	DMB	15.1	568		2	
DBD@X02A	DMB	15.1	416		1	
DBD@X03A	DMB	15.1	416		1	
DBD@X04A	DMB	15.1	416		2	
DBD@X05A	DMB	15.1	416		1	
DBD@X06A	DMB	15.1	568		2	ALIAS
DBD@X07A	DMB	15.1	416		1	
DBD@X08A	DMB	15.1	416		1	

Figure 205. Library Contents report in the SYSPRINT data set (Part 1 of 2)

+-----+  
 |LIBRARY CONTENTS|  
 +-----+

PSB DETAIL  
 =====

MBRNAME	TYPE	IMS LVL	SIZE (BYTES)	# DMBS REFERENCE	# PSBS REFERENCNG	COMMENTS
PSB@001	PSB	15.1	10,944	16		
PSB@002	PSB	15.1	4,944	8		
PSB@005	PSB	15.1	288	0		NO DBDS REFERENCED
PSB@007	PSB	15.1	512	1		

SUMMARY  
 =====

TOTAL: DMBS = 21                      PSBS = 4                      #REF\_DMBS = 25

Figure 206. 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.

+-----+  
 |LIBRARY CONTENTS|  
 +-----+

DMB DETAIL  
 =====

MBRNAME	TYPE	IMS LVL	SIZE (BYTES)	# DMBS REFERENCE	# PSBS REFERENCNG	COMMENTS	ACBGEN	
							DATE	TIME
DBD@D01A	DMB	15.1	648		1		09/13/2021	16:06:28.77
DBD@D03A	DMB	15.1	1,192		1		09/13/2021	16:06:28.77
DBD@E01A	DMB	15.1	1,336		1		09/13/2021	16:06:28.77
DBD@E02A	DMB	15.1	2,504		1		09/13/2021	16:06:28.77
DBD@H01A	DMB	15.1	600		1		09/13/2021	16:06:28.77
DBD@H02A	DMB	15.1	1,344		2		09/13/2021	16:06:28.77
DBD@ISAM	DMB	15.1	560		1		09/13/2021	16:06:28.77
DBD@I01A	DMB	15.1	680		1		09/13/2021	16:06:28.77
DBD@I02A	DMB	15.1	392		1		09/13/2021	16:06:28.77
DBD@I03A	DMB	15.1	872		1		09/13/2021	16:06:28.77
DBD@M02A	DMB	15.1	200		1		09/13/2021	16:06:28.77
DBD@M04A	DMB	15.1	120		1		09/13/2021	16:06:28.77
DBD@S02A	DMB	15.1	616		1		09/13/2021	16:06:28.77
DBD@X01A	DMB	15.1	568		2		09/13/2021	16:06:28.77
DBD@X02A	DMB	15.1	416		1		09/13/2021	16:06:28.77
DBD@X03A	DMB	15.1	416		1		09/13/2021	16:06:28.77
DBD@X04A	DMB	15.1	416		2		09/13/2021	16:06:28.77
DBD@X05A	DMB	15.1	416		1		09/13/2021	16:06:28.77
DBD@X06A	DMB	15.1	568		2	ALIAS	09/13/2021	16:06:28.77
DBD@X07A	DMB	15.1	416		1		09/13/2021	16:06:28.77
DBD@X08A	DMB	15.1	416		1		09/13/2021	16:06:28.77

Figure 207. Library Contents report in the FABQRPT data set (Part 1 of 2)

+-----+  
 |LIBRARY CONTENTS|  
 +-----+

PSB DETAIL  
 =====

MBRNAME	TYPE	IMS LVL	SIZE (BYTES)	# DMBS REFERENCE	# PSBS REFERENCNG	COMMENTS	ACBGEN	
							DATE	TIME
PSB@001	PSB	15.1	10,944	16			09/13/2021	16:06:28.77
PSB@002	PSB	15.1	4,944	8			09/13/2021	16:06:28.77
PSB@005	PSB	15.1	288	0		NO DBDS REFERENCED	09/13/2021	16:06:28.77
PSB@007	PSB	15.1	512	1			09/13/2021	16:06:28.77

SUMMARY  
 =====

TOTAL: DMBS = 21      PSBS = 4      #REF\_DMBS = 25

Figure 208. Library Contents report in the FABQRPT data set (Part 2 of 2)

## Report field descriptions

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.

**Note:** Even when the IMS installed level is updated to 15.2 and ACB is generated with IMS, IMS 15.1 is still used for such ACB.

### 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 REFERENCNG

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

The Distribution of Member Sizes 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.

```
IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE: 5
5655-U08                DATE: 10/01/2021  TIME: 17.07.07        FABQCHECK - V2.R2

+-----+
|DISTRIBUTION OF MEMBER SIZES|
+-----+

          DMBs
          =====

          RANGE                COUNT
          -----                -
          0 - 511                9
          512 - 1,023            8
          1,024 - 1,535          3
          2,048 - 2,559          1

          TOTAL BYTES =         14,696

          PSBs
          =====

          RANGE                COUNT
          -----                -
          0 - 1,023              2
          4,096 - 5,119          1
          10,240 - 11,263        1

          TOTAL BYTES =         16,688
```

Figure 209. Distribution of Member Sizes report

## Distribution of PSB Workarea Sizes report

The Distribution of PSB Workarea Sizes report helps you determine the size of the buffer pool that holds PSBs in the online environment. The distribution interval is in 1K 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.

```
IMS LIBRARY INTEGRITY UTILITIES - ADVANCED ACB GENERATOR          PAGE:      6
5655-U08                DATE: 10/01/2021  TIME: 17.07.07        FABQCHECK - V2.R2

+-----+
|DISTRIBUTION OF PSB WORKAREA SIZES|
+-----+

          RANGE                COUNT
-----
          0 - 1,023             2
        1,024 - 2,047           2

TOTAL BYTES =                4,080
```

Figure 210. Distribution of PSB Workarea Sizes report

## Chronological History of ACBGENs report

The Chronological History of ACBGENs report provides 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” on page 364](#)
- [“Report field descriptions” on page 365](#)

### Sample report

The following figure shows an example of the Chronological History of ACBGENs report.

+-----+  
 |CHRONOLOGICAL HISTORY OF ACBGENS|  
 +-----+

DATE	DATE	TIME	#PSB'S	#DMB'S
2021.021	01/21/2021	17:29:55	1	0
2021.025	01/25/2021	09:03:22	0	1
2021.032	02/01/2021	11:45:42	1	0
2021.079	03/20/2021	16:32:57	1,177	6
2021.079	03/20/2021	19:29:34	0	1
2021.082	03/23/2021	13:39:33	1	0
2021.089	03/30/2021	10:27:19	6	0
2021.090	03/31/2021	17:36:48	3	0
2021.101	04/11/2021	10:23:03	1	0
2021.101	04/11/2021	10:34:28	0	1
2021.102	04/12/2021	13:27:51	0	1
2021.103	04/13/2021	09:43:54	1	0
2021.105	04/15/2021	07:07:41	0	1
2021.108	04/18/2021	20:32:42	1	0
2021.108	04/18/2021	20:38:53	1	0
2021.108	04/18/2021	20:43:21	1	0
2021.111	04/21/2021	15:18:03	1	0
2021.111	04/21/2021	15:29:47	1	0
2021.124	05/04/2021	13:19:57	1	0
2021.124	05/04/2021	13:51:18	6	0

Note: portions of this report omitted

2021.188	07/07/2021	13:59:04	1	0
2021.266	09/23/2021	15:26:58	0	6
2021.267	09/24/2021	11:37:01	6	0

NUMBER OF UNIQUE ACBGEN TIMESTAMPS = 140

Figure 211. 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

The Warning Messages 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.

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 'BE4LORDR' REFERENCED BY PSB 'PE4YOINQ' IS NOT IN ACBLIB
FABQ1031W DMB 'BE4LPART' REFERENCED BY PSB 'PE4CPPUR' IS NOT IN ACBLIB
FABQ1031W DMB 'C330INVC' REFERENCED BY PSB 'C330READ' IS NOT IN ACBLIB
FABQ1031W DMB 'DLCDR01 ' REFERENCED BY PSB 'VS1CDRMS' IS NOT IN ACBLIB
```

*Figure 212. Warning Messages report*



---

## Chapter 13. 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 367](#)
- [“Restrictions and considerations for the MFS Reversal utility” on page 368](#)
- [“Converting MFS control blocks to control statements” on page 368](#)
- [“JCL requirements for the MFS Reversal utility” on page 368](#)
- [“Control statements for the MFS Reversal utility” on page 370](#)
- [“Output from the MFS Reversal utility” on page 373](#)
- [“Important notes about the generated source” on page 379](#)

---

### 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” on page 367](#)
- [“Program structure” on page 367](#)

#### 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 379](#) 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 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 368](#).
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 370](#).
3. Submit the job.
4. Check the output data sets that are generated.  
See [“Output from the MFS Reversal utility” on page 373](#).

### 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 379](#) 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” on page 369](#)
- [“EXEC statement” on page 369](#)
- [“DD statements” on page 369](#)

## 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
// DD DSN=IMSVS.SDFSRESL,DISP=SHR
//SYSPRINT DD SYSOUT=*,DCB=BLKSIZE=133
//SYSOUT DD SYSOUT=*,DCB=BLKSIZE=133
//FORMAT DD DSN=ims.format,DISP=SHR
//MFSSRCE DD DSN=user.mfs.source,DISP=SHR
//SYSIN DD *
```

Figure 213. 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 370](#).

### **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*.

### **COPYRPT 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” on page 370](#)
- [“Control statement keywords” on page 370](#)

### **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 DFSUDT0*x*. 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:

- MIDs and MODs that are referenced by the MIDs and MODs that are specified on the SELECT statement
- DIFs and DOFs that are associated with the MIDs and MODs that are specified on the SELECT statement
- MIDs 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 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
```

**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 COPYPRT 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.

```
IMS LIBRARY INTEGRITY UTILITIES - MFS REVERSAL/COMPARE          "MESSAGES"          PAGE: 00001
5655-U08              DATE: 10/01/2021  TIME: 16.16.51          FABVLOG - V2.R2

DSNAME: IMSVS.FORMAT

FABV0022W USING DEFAULT DEVICE CHARACTERISTICS TABLE FABVDVCT
FABV0044I SOURCE FOR MEMBER SAMFMX   BUILT
```

Figure 214. Messages 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” on page 373](#)
- [“Report field descriptions” on page 374](#)

### Sample report

The following figure shows an example of the MID/MOD XREF report.

DSNAME: IMSVS.FORMAT

MSGNAME	SOR=	DCODE	DEVICE	FCODE	FEATURE	NXT=	REFERENCED BY
MO3270B	(O) F3270B	02	3270,2	7F	FEAT=IGNORE		
MI3270K	(I) F3270K	42	3270-A02	C7	FEAT=(CARD,PFK,PEN)	MO3270K	MO3270K
MO3270K	(O) F3270K	42	3270-A02	C7	FEAT=(CARD,PFK,PEN)	MI3270K	MI3270K
MO3270J	(O) F3270K	42	3270-A02	C7	FEAT=(CARD,PFK,PEN)		MI3270K
MI360B	(I) FI360B	08	FIN	7F	FEAT=IGNORE	MO3270K	MI360B
MI7108	(I) DI7108	0C	SCS1	01	FEAT=1		MI3270K
MO3270C	(O) F3270C	02	3270,2	7F	FEAT=IGNORE	MI3270C	

WARNING:  
THE FOLLOWING MSG FORMATS REFERENCE DEVICE FORMATS FOR WHICH SOURCE WAS CREATED. HOWEVER THESE MSG FORMAT NAMES WERE NOT INCLUDED ON THE UTILITY SELECT STATEMENT AND THEREFORE SOURCE FOR THEM WAS NOT GENERATED. ALSO THE CROSS REFERENCE INFORMATION BETWEEN MID AND MOD ARE NOT PRINTED.

MI3270C	(I) F3270C	02	3270,2	7F	FEAT=IGNORE
MOSCS1C	(O) F3270C	02	3270,2	7F	FEAT=IGNORE

Figure 215. 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 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.



- 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

The MFSSRCE data set, which is a 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” on page 375](#)
- [“Naming conventions” on page 375](#)
- [“Example of created source” on page 376](#)

### 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 DPAXxxxx, where xxxxx 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 DLnnnnnn, where nnnnnn is the decimal equivalent of an internal offset.

#### CURSOR name

Each cursor name is of the form CSnnnnnn, where nnnnnn 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 COND*mmmm*, where *mmmm* 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 xxxxxx 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 NN000SSC.

**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 ACTVPIDdd, 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 IFttttt, where ttttt 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 M03270A,MI3270A
```

In this example, the MFS Reversal utility encountered a member X'027F'F3270A 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 527 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 379 for additional information about the generated source.

The following figure shows an example of the source that is created by MFS Reversal.

```

F3270A   FMT
        DEV   TYPE=(3270,2),
              FEAT=IGNORE,
              DSCA=X'0230',
              SYMSG=DL000022
        DIV   TYPE=INOUT
DPA00001 DPAGE FILL=NONE,
              CURSOR=((5,17,CS000030))
        DFLD  'MFS SUN SPECIFICATIONS',
              POS=(2,28),
              EATTR=(HBLINK,RED,PX'41'),
              ATTR=(PROT)

DL000006 DFLD  POS=(5,57),
              LTH=15,
              EATTR=(VMFLD,HUL,RED,PX'42',RIGHT,OVER,LEFT),
              ATTR=(NUM,MOD)
DL000022 DFLD  POS=(24,2),
              LTH=79,
              ATTR=(PROT,NUM)
*-----*
          FMTEND
*-----*
M03270A  MSG   TYPE=OUTPUT,
              SOR=(F3270A,IGNORE),
              OPT=2,
              NXT=MI3270A,
              FILL=PT
        LPAGE SOR=DPA00001,
              COND=(COND0005,=,'CA'),
              NXT=MI3270A
        SEG
        MFLD  DL000010,
              LTH=30,
              ATTR=YES
COND0005 MFLD  DL000020,
              LTH=2
        MSGEND
*-----*
MI3270A  MSG   TYPE=INPUT,
              SOR=F3270A,
              OPT=2
        LPAGE SOR=DPA00001,
              NXT=M03270A
        SEG   EXIT=(127,0)
        MFLD  'SUNTRANA '
        MFLD  DL000012,
              LTH=5
        MSGEND
*-----*
          END

```

Figure 216. Sample Source 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” on page 378](#)
- [“Sample report: OPTION COPY=YES,REPLACE=YES” on page 378](#)

## 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.

```
IMS LIBRARY INTEGRITY UTILITIES - MFS REVERSAL          "COPY REPORT"          PAGE: 00001
5655-U08          DATE: 10/01/2021  TIME: 12.10.56          FABVRVRS - V2.R2

COPY TO DSNAME: IMSVS.CPYFMT
NUMBER OF COPIED MEMBERS =          16
NUMBER OF COPY SKIPPED MEMBERS =      12

THE FOLLOWING MEMBERS WERE COPIED
-----
"FM100  "FM100  "FMT762  "FMT762  MIDA  MIDA2  MIDB  MIDB2  MIDC2  MIDD2
MODA   MODA2   MODB   MODB2   MODC2 MODD2
THE FOLLOWING MEMBERS WERE SKIPPED
-----
"FM122  "FM123  "FMT122  "FMT123  MID1003  MID1004  MID1005  MID1006  MOD1003  MOD1004
MOD1005  MOD1006
```

Figure 217. Copy report (OPTION COPY=YES,REPLACE=NO)

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.

```
IMS LIBRARY INTEGRITY UTILITIES - MFS REVERSAL          "COPY REPORT"          PAGE: 00001
5655-U08          DATE: 10/01/2021  TIME: 12.10.56          FABVRVRS - V2.R2

COPY TO DSNAME: IMSVS.CPYFMT
NUMBER OF COPIED MEMBERS =          16
NUMBER OF COPY FAILED MEMBERS =        8

THE FOLLOWING MEMBERS WERE COPIED
-----
"FM100  "FM100  "FMT762  "FMT762  MIDA  MIDA2  MIDB  MIDB2  MIDC2  MIDD2
MODA   MODA2   MODB   MODB2   MODC2 MODD2
THE FOLLOWING MEMBERS COULD NOT BE COPIED
-----
"FM210  "FM220  "FMT210  "FMT220  MID2001  MID2002  MOD2001  MOD2002
```

Figure 218. Copy report (when copy failed for some members)

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.

```

COPY TO DSNAME: IMSVS.CPYFMT
NUMBER OF COPIED MEMBERS      =      16
NUMBER OF REPLACED MEMBERS    =      12

THE FOLLOWING MEMBERS WERE COPIED
-----
" "FMT100      " "FMT100      " "FMT762      " "FMT762      MIDA      MIDA2      MIDB      MIDB2      MIDC2      MIDD2
MODA          MODA2          MODB          MODB2          MODC2          MODD2
THE FOLLOWING MEMBERS WERE REPLACED
-----
" "FMT122      " "FMT123      " "FMT122      " "FMT123      MID1003    MID1004    MID1005    MID1006    MOD1003    MOD1004
MOD1005      MOD1006
    
```

Figure 219. Copy report (OPTION COPY=YES,REPLACE=YES)

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'OE' 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 ENDMSG or (BGNMSG,ENDMSG) or (BGNPP,BGNMSG,ENDMSG) is the EJECT option, the ENDMSG 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 14. 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 381](#)
- [“Considerations for the MFS Compare utility” on page 382](#)
- [“Keywords used in comparisons” on page 382](#)
- [“Comparing MFS control blocks” on page 386](#)
- [“JCL requirements for the MFS Compare utility” on page 387](#)
- [“Control statements for the MFS Compare utility” on page 389](#)
- [“Output from the MFS Compare utility” on page 390](#)

---

### 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” on page 381](#)
- [“Program structure” on page 382](#)

#### 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

### **SYMSG**

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 separator 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

**ACTVPID**

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

### **SYMSG**

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 387.

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 389.

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 390.

## 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”](#) on page 387
- [“EXEC statement”](#) on page 387
- [“DD statements”](#) on page 387

### JCL example

An example of the JCL that is required for MFS Compare is shown in the following figure.

```
//stepname EXEC PGM=FABVCMR
//STEPLIB DD DSN=HPS.SHPSLMDO,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))
//SYSIN DD *
```

Figure 220. Example of MFS Compare JCL (FABVCMR JCL)

Modify the fields shown in lower case to reflect your operating environment.

### EXEC statement

This statement invokes the MFS Compare utility, FABVCMR. The statement must be in the following format:

```
//stepname EXEC PGM=FABVCMR
```

### 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 FABVCMR 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.

**SYSRINT 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.

**SYSYOUT DD**

This sequential data set contains activity logs and any errors encountered during the execution of the FABVCMR 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 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.

**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 389.

## 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” on page 389](#)
- [“Control statement keywords” on page 389](#)

### 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 DFSUDT0*x*. 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.

```
IMS LIBRARY INTEGRITY UTILITIES - MFS REVERSAL/COMPARE          "MESSAGES"          PAGE: 00001  
5655-U08              DATE: 10/01/2021  TIME: 16.03.02      FABVLOG - V2.R2  
  
DSNAME: IMSVS.FORMAT1  
  
FABV0022W USING DEFAULT DEVICE CHARACTERISTICS TABLE FABVDVCT  
FABV0044I SOURCE FOR MEMBER SAMFMX   BUILT  
IMS LIBRARY INTEGRITY UTILITIES - MFS REVERSAL/COMPARE          "MESSAGES"          PAGE: 00002  
5655-U08              DATE: 10/01/2021  TIME: 16.03.02      FABVLOG - V2.R2  
  
DSNAME: IMSVS.FORMAT2  
  
FABV0022W USING DEFAULT DEVICE CHARACTERISTICS TABLE FABVDVCT  
FABV0044I SOURCE FOR MEMBER SAMFMX   BUILT
```

Figure 221. Messages in the SYSOUT data set

### 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” on page 390](#)
- [“Report field descriptions” on page 391](#)

### Sample report

The following figure shows an example of the MFS Compare report.



```

1)          FORMAT: VNDR400.FORMAT          FORMAT2: VNDR400.FORMATC
2) LABEL    STATEMENT  KEYWORD              LABEL    STATEMENT  KEYWORD
3) ADDFMT   FMT                NOT IN LIBRARY
4) CDCPI1   FMT                -----
5) FISC1A   FMT                -----
6) DEV0C7F DEV                -----
7) DIVIN    DIV                -----
8) DPA00002 DPAGE              -----
9)          DL000034   DFLD   POS=(1,13)
10)         DL000034   DFLD   LTH=5
11) MISC1A   MSG                -----
12)         -----
13) FISC1B   FMT                NOT IN LIBRARY
14) F0SC1A   FMT                -----
15) DEV0C7F DEV   FORMS='SCS1A.OUT'      DEV0C7F   DEV   FORMS='SCS1B.OUT'
16) DEV0C7F DEV   XTAB=(OFFLINE,5,HT=(8,10,12))  DEV0C7F   DEV   HTAB=(OFFLINE,5,HT=(
17)         8,10,12))
18) M0SC1A   MSG                -----
19)         LPAGE
20)         SEG
21)         MFLD   NN001079              MFLD   NN001001
22)         MFLD   NN001080              MFLD   NN001002
23)         LPAGE
24)         SEG
25)         MFLD   NN001081              MFLD   NN001003
26)         -----
27)         -----

```

Figure 222. 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.

FORMAT2 is the DD name as required by the JCL, and VNDR400.FORMAT2 is the data set name of the second format library whose members are to be compared.

### Line 2

This header line specifies a LABEL STATEMENT and KEYWORD sequence for each format library.

On subsequent lines, if an FMT, FCB, field name, literal or keyword of a parameter is present under a library, it indicates that the parameter from that library

- is either in error, or
- is present, but the corresponding parameter from the other library is absent.

### 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 527.

### 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 15. 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 393](#)
- [“IMS Library Integrity Utilities abend codes” on page 399](#)
- [“IMS messages” on page 400](#)
- [“IMS Library Integrity Utilities messages” on page 400](#)
- [“How to look up message explanations” on page 519](#)
- [“Gathering diagnostic information” on page 520](#)
- [“Diagnostics Aid” on page 520](#)

---

## 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 25. LICON utility return codes*

---

<b>Return code</b>	<b>Meaning</b>
0	Successful completion. The program ended successfully.
4	Warning message. Warning messages were issued, but the requested operation was completed.
8	Error message. Error messages were issued, and requested operations for some (but not all) databases failed.
16	Error message. Error messages were issued. <ul style="list-style-type: none"><li>• No databases or RDEs were successfully processed.</li><li>• A severe error occurred.</li></ul> The LICON utility ended immediately after the error condition detected. It skips processing any subsequent commands.

---

## Consistency Checker return codes

The Consistency Checker utility returns one of the four return codes.

*Table 26. Consistency Checker utility return codes*

<b>Return code</b>	<b>Meaning</b>
0	Successful completion.  The program ended successfully. If, for any reason, reports are not produced, there is an explanation in the activity log.
4	Warning message.  Warning messages were issued, but the requested operation was completed.
8	Error message.  Error messages were issued, and some requested operations were skipped or ended unsuccessfully because severe errors were detected.  When the consistency check of a DBD or a PSB fails, the program returns a code of 8 as the default. If other return code is specified with the FAILRC keyword in the SYSIN data set, the program returns the specified code.
12	Error message.  Error messages were issued. The requested operations were not performed because the control statement errors were detected or the specified DFSRESLB IMS version was not supported.

## Multiple Resource Checker return codes

The Multiple Resource Checker utility returns one of the three return codes.

*Table 27. Multiple Resource Checker utility return codes*

<b>Return code</b>	<b>Meaning</b>
0	Successful completion.  The program ended successfully. If, for any reason, reports are not produced, there is an explanation in the activity log.
4	Warning message.  Warning messages were issued, but the requested operation was completed.
8	Error message.  Error messages were issued, and some requested operations were skipped or ended unsuccessfully because severe errors were detected.

## DBD/PSB/ACB Compare, Mapper, and Reversal return codes

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 return one of the four return codes.

*Table 28. DBD/PSB/ACB Compare, Mapper, and Reversal utility return codes*

<b>Return code</b>	<b>Meaning</b>
0	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.
4	Warning message. Warning messages were issued, but the requested operation was completed.
8	Unsuccessful execution. Some requested operations were skipped or executed unsuccessfully because severe errors were detected.
16	Error message. Error messages were issued. The requested operations were not performed because the control statement errors were detected.

## MDA Reversal return codes

The MDA Reversal utility returns one of the three return codes.

*Table 29. MDA Reversal return codes*

<b>Return code</b>	<b>Meaning</b>
0	Successful completion. The program ended successfully.
4	Warning message. Warning messages were issued, but the requested operation was completed.
8	Error message. Error messages were issued, and some requested operations were skipped or ended unsuccessfully because severe errors were detected.

## Catalog Manager return codes

The Catalog Manager utility returns one of the three return codes.

Table 30. Catalog Manager utility return codes

Return code	Meaning
0	Successful completion. The program ended successfully.
4	Warning message. Warning messages were issued, but the requested operation was completed. Or, the utility skipped processing the failed members and error messages were issued, but the requested operation was completed.
8	Error message. Error messages were issued, and some requested operations were skipped or ended unsuccessfully because severe errors were detected.

## Advanced ACB Generator return codes

Whenever an error condition is detected by the Advanced ACB Generator utility, 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

The MFS Reversal utility returns one of the following return codes.

Table 31. MFS Reversal utility return codes

Return code	Meaning
0	Successful completion. The program ended successfully.
4	Warning message. Warning messages were issued, but the requested operation was completed.
8 or higher	Error message. Error messages were issued. The program ended unsuccessfully.

## MFS Compare return codes

The MFS Compare utility returns one of the following return codes.

*Table 32. MFS Compare utility return codes*

<b>Return code</b>	<b>Meaning</b>
0	Successful completion. The program ended successfully.
8 or higher	Either of the following errors occurred: <ul style="list-style-type: none"><li>• The program found differences between MFS control blocks.</li><li>• Error messages were issued. The program ended unsuccessfully.</li></ul>

## IMS Library Integrity Utilities return codes under IMS Administration Tool

The return codes of IMS Library Integrity Utilities when the utility is used under IMS Administration Tool are explained in this topic.

### 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 33. IMS Library Integrity Utilities return codes under IMS Administration Tool*

<b>Return code</b>	<b>Meaning</b>
0	Successful completion. The program ended successfully.
2	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.
4	Warning message. Warning messages were issued, but the requested operation completed. <ul style="list-style-type: none"><li>• One or more differences or mismatches were detected during the compare operation.</li><li>• The specified PSB is not valid for IMS SQL.</li><li>• The DBD or PSB resource was not found.</li><li>• DBD instances with an old DB Version are not used because database versioning is not enabled.</li></ul>
8	Error message. Error messages were issued, and the requested operation failed.

---

Table 33. IMS Library Integrity Utilities return codes under IMS Administration Tool (continued)

---

<b>Return code</b>	<b>Meaning</b>
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.

---

### **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 34. FABXAEXP return codes

---

<b>Return code</b>	<b>Meaning</b>
0	Successful completion. The program ended successfully.
2	Warning message. Warning messages were issued while decoding one ore 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. The DBD or PSB resource was not found.
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.

---



---

Table 34. FABXAEXP return codes (continued)

---

<b>Return code</b>	<b>Meaning</b>
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 U001, U002, U003, U004, and U005. 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 Library Integrity Utilities abend codes when run under other tools to write DBD and PSB maps

IMS Library Integrity Utilities which is run under other tools to write DBD and PSB maps uses abend codes U3901, U3902, U3903, U3904, U3905, and U3909. Before issuing the abend code, the utility always writes a message that identifies the problem.

## 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:

### FABx

Indicates that the message was issued by IMS Library Integrity Utilities. x is one of L, M, N, Q, V, W, and X.

### L

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.

### M

Indicates that the message was issued by the DBD/PSB/ACB Mapper utility.

### N

Indicates that the message was issued by the DBD/PSB/ACB Reversal utility or the Reversal Site Default Generation utility.

### Q

Indicates that the message was issued by the Advanced ACBGEN utility.

### V

Indicates that the message was issued by the MFS Reversal utility or the MFS Compare utility.

### W

Indicates that the message was issued by the Multiple Resource Checker utility.

### X

Indicates that the message was issued by the Catalog Manager utility or issued while using the DBD/PSB Map Viewer.

### nnnn

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. Also, some messages that are issued when you use the Catalog Manager utility or when Library Integrity Utilities is run under IMS Administration Tool also begin with the prefix FABL.

---

**FABL0001I      CONTROL CARD SUPPLIED IS:  
                  echo of control statement**

**System action**

The DBD/PSB/ACB Compare utility skips this control statement and continues processing.

**Explanation**

This message is the echo of the SYSIN control statements that are checked by this utility.

**User response**

Correct the format of this control statement, and rerun the job.

**System action**

The DBD/PSB/ACB Compare utility continues processing.

---

**FABL0003E      LIBRARY MISSING FOR [DBD |  
                  PSB | ACB] COMPARE**

**Explanation**

The specification of the libraries required to execute the function is missing or invalid.

**User response**

None. This message is informational.

---

**FABL0002W      INVALID STATEMENT IN SYSIN  
                  DATASET**

**System action**

Skips the reporting process for this function.

**Explanation**

A control statement with invalid format was found in the SYSIN data set.

**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 | IMS DIRECTORY OF INPUTx]

### Explanation

The specified member, or one or more of the members specified by a wildcard, were not found in the DBD/PSB/ACB library or in the IMS directory. 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 *acclib\_ddname*

### Explanation

An error occurred in the reading of an ACB member in the *acclib\_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=*x.x*****Explanation**

The ACB member generated by IMS version *x.x* 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: DEDB, 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.

The compare function of the Catalog Manager utility ends with a return code of 04.

## 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 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 *dbdname* 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. *dbdname* 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.

---

**FABL0032W      DBD *dbdname* FOR LOGICAL IS  
                  NOT COMPARED.**

## Explanation

The Catalog Manager utility does not compare the logical DBD because there are no ACBs for a logical database in the IMS directory.

## System action

The Catalog Manager utility skips this member and continues processing.

## User response

None.

---

**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 | IMS DIRECTORY]

## 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 or in the IMS directory. 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  
*member1:member2*

## 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.

Detectable DBD type	Field	Description
DBD and DBD TYPE ACB	DMBDASZE	The size of RDMVTAB CSECT
DBD and DBD TYPE ACB	DMBDAKL	The executable key length of root
DBD	DMBDANME	The name of randomizer module
DBD	DMBDARAP	The number of root anchor points or blocks
DBD	DMBDABLK	The number of the highest blocks that are directly addressed
DBD	DMBDABYM	The maximum number of bytes

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** COMPARED *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:

Control statement	Result
DBD=DBD@D01A : DBD@D02A	Different
DBD=DBD@D03A	Match
PSB=PSB@001A : PSB@002A	Different
ACB=DBD@D01A	Different

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** COMPARED *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\_source1* shows the total number of DBDs, PSBs, or ACBs that were compared at their source levels.
- *number\_of\_source2* shows the number of DBDs, PSBs, or ACBs that are identical at their source levels.
- *number\_of\_source3* 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:

Control statement with REPORT=SOURCE	Result
DBD=DBD@D01A : DBD@D02A	Different
DBD=DBD@D03A	Match
PSB=PSB@001A : PSB@002A	Different
ACB=DBD@D02A	Different
DBD=DBD@D01A , ACB	Different
PSB=PSB@001A , ACB	Match
ACB=DBD@D01A , DBD	Different
ACB=PSB@001A , PSB	Match

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.

---

**FABL0043W THE NODIFF PARAMETER IS INVALID WITHOUT THE SOURCE PARAMETER**

### Explanation

The NODIFF parameter is ignored. The NODIFF parameter is used only when the SOURCE parameter is specified in the REPORT statement.

### System action

The DBD/PSB/ACB Compare utility skips the NODIFF parameter and continues processing.

### User response

To generate a source-level compare report even when no difference is found, add REPORT=SOURCE to the control statement, and rerun the job.

---

**FABL0044I CONTROL STATEMENT REPORT IS WRITTEN TO SYSPRINT**

### Explanation

The control statement report is written to the SYSPRINT data set.

### System action

The DBD/PSB/ACB Compare utility continues processing.

### User response

None. This message is informational.

---

**FABL0045W DYNAMIC ALLOCATION FAILED FOR DSNAME=*data\_set\_name*. RETURN CODE=*return\_code*, REASON CODE=*reason\_code***

## Explanation

An attempt to dynamically allocate the indicated data set failed. *return\_code* is the hexadecimal return code, and *reason\_code* is the hexadecimal reason code.

## System action

The DBD/PSB/ACB Compare utility continues processing.

## User response

This error is likely an internal system error. Collect the dump, and contact IBM Software Support.

---

**FABL0046W THE CONTROL STATEMENT REPORT IS NOT PRINTED BECAUSE DYNAMIC ALLOCATION FAILED**

## Explanation

The control statement report is not written to the SYSPRINT data set because the dynamic allocation of a work data set failed.

## System action

The DBD/PSB/ACB Compare utility continues processing.

## User response

Locate the preceding message FABL0045W, which describes the error condition. Correct the error, and rerun the job.

---

**FABL0047W GSAM DBD *dbdname* 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. *dbdname* is the DBD member name that was specified.

## System action

IMS Library Integrity Utilities continues processing.

## User response

None. This message is informational.

---

**FABL0048E 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 *IMS Tools Base: IMS Tools Common Services User's Guide and Reference*. If necessary, correct the error condition and rerun the job.

---

**FABL0049I COMPARED *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.

---

**FABL0051W      VERSION AND EXIT PARAMETERS ARE NOT COMPARED FOR DBD *dbdname*. 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.

---

**FABL0052E      IMS DIRECTORY SPECIFIED IN INPUTx IS EMPTY.**

## Explanation

The Catalog Manager utility found that the IMS directory that is specified by the INPUTx keyword is empty. The IMS directory contains no members to compare.

## System action

The Catalog Manager utility ends with a return code of 8.

## User response

Ensure that the IMS directory specified on the INPUTx keyword is correct.

---

**FABL0053W      VENDOR SECTION IS NOT COMPARED**

## Explanation

While processing DBD or PSB, the DBD/PSB/ACB Compare utility detected a vendor section. This section is not compared.

## System action

The DBD/PSB/ACB Compare utility continues processing.

## User response

None.

---

**FABL0054W      VERSION AND EXIT PARAMETERS ARE NOT COMPARED FOR DBD *dbdname*.**

## Explanation

The following parameters are not compared because these parameters could not be obtained from the PSB-type ACB member that refers to the reported DBD-type ACB member.

- The VERSION parameter of the DBD statement
- The EXIT parameter of the DBD and SEGM statements

## System action

IMS Library Integrity Utilities continues processing.

## User response

None.

---

**FABL0055W      SENSEG STATEMENT FOR MSDB IS NOT DECODED FOR PSB *member*. NO MSDB IS REFERRED BY PSB IN IMS DIRECTORY STAGING DATASET.**

## Explanation

The parameter is not decoded because this parameter could not be obtained from the MSDB member.

## System action

IMS Library Integrity Utilities continues processing.

## User response

None.

---

**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.

---

**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 FABLRTx 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.n***

## Explanation

The Integrity Checker utility was run under an unsupported version of IMS. *nn.n* 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:  
*ddddddd [AREA: aaaaaaaaa]***

## Explanation

The registered DMB entry (RDE) has been successfully created for non-HALDB full-function database or HALDB partition *ddddddd*, or area *aaaaaaaa* of database *ddddddd*.

## 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 (*dddddddd*) 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 (*dddddddd*) 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 with the reason code \$\$.

### User response

If the mismatch is unexpected, check whether the DBD, ACB, or RECON that you are using is correct.

---

**FABL0205E**      **VERIFICATION PROCESS FOR**  
**ddddddd [, aaaaaaaaa] 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 doing 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 [, aaaaaaaaa] 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.

---

<b>FABLO207E</b>	<b>RDE CREATION FAILED FOR DATABASE: <i>dddddddd</i> [AREA: <i>aaaaaaaa</i>]</b>
------------------	--

### 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.

---

<b>FABLO208W</b>	<b>CHECKSUM MISMATCH FOUND FOR DBD: <i>dddddddd</i> [AREA: <i>aaaaaaaa</i>]</b>
------------------	---

### 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 FABLO208W messages are issued. Other FABLO208W 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 of the exit routine.

### User response

If the mismatch is unexpected, check whether the exit routine you are using is correct.

---

<b>FABLO209E</b>	<b>CHECKSUM MISMATCH FOUND FOR DBD: <i>dddddddd</i> [AREA: <i>aaaaaaaa</i>]</b>
------------------	---

### 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 FABLO209E messages are issued. Other FABLO209E 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.

---

<b>FABLO210E</b>	<b>MODULE NOT FOUND FOR <i>module</i></b>
------------------	---

### 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: aaaaaaaaa]**

## 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 *ddddddd* or area *aaaaaaaa* of database *ddddddd*.

## 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 *ddddddd* or area *aaaaaaaa* of database *ddddddd*, 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 *ddddddd* or area *aaaaaaaa* of database *ddddddd*.

## 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 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.

---

**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 (IARR2V-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: *aaaaaaaa*]**

## 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:**  
*ddname*

## Explanation

The SNAP macro issued to data set whose DD name is *ddname* failed.

## System action

Processing continues without using the indicated data set.

## User response

Check the status of the data set *ddname*. 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 *ddddddd* 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 *ddddddd* 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'tttrrc'**  
**RC=rr**

## Explanation

During the ACB member processing, a read error was detected for the ACB member whose TTRC is *x'tttrrc'*. *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. Check whether the statement continuation is correctly completed. 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.

---

**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      PARENTHESIS 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: *tttttttttttt***

## Explanation

The specified time stamp value *tttttttttttt* is incorrect. The correct format is *YYYYDDHMMSSSTT*, which identifies one of the expired RDEs.

## 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.

---

**FABL0431E NO OPTION VALUE FOUND**

## Explanation

No option value is supplied in the parenthesis.

## 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.

---

**FABL0432E THE ORDER OF THE PARAMETER FOR THE IMS CATALOG IS INCORRECT.**

## Explanation

The order of the parameter for the IMS catalog is incorrect. It must be specified at the top of the FABLIN control statement.

## System action

The LICON utility ends immediately with a job step return code of 16.

## User response

See “Runtime options” on page 88 and correct the order of the parameter, and then rerun the utility.

---

**FABL0433E INCORRECT VALUE IS SPECIFIED FOR [IMSCAT|IMSCATHLQ]**

## Explanation

The specified value for the indicated parameter is incorrect.

## System action

The LICON utility ends immediately with a job step return code of 16.

## User response

See “LICON utility reference” on page 85 and correct the value, and then rerun the utility.

---

**FABL0440I DATABASE: dddddddd [AREA: aaaaaaaaa] SUCCESSFULLY PROCESSED**

## Explanation

This message shows that non-HALDB full-function database or HALDB partition *ddddddd* or area *aaaaaaaa* of database *ddddddd* has been successfully processed by the LICON utility command.

## System action

Processing continues.

## User response

None. This message is informational.

---

**FABL0441I DATABASE: dddddddd [AREA: aaaaaaaaa] TIMESTMP: tttttttttttt SUCCESSFULLY PROCESSED**

## Explanation

The message shows that the RDE for non-HALDB full-function database or HALDB partition *ddddddd* or area *aaaaaaaa* of database *ddddddd* and the time stamp *tttttttttttt* has been successfully processed by the LICON utility command.

## System action

Processing continues.

## User response

None. This message is informational.

---

**FABL0442E CURRENT RDE FOR DATABASE: dddddddd [AREA: aaaaaaaaa] NOT REPLACED BECAUSE NO REPLACE OPTION SPECIFIED**



## Explanation

The INIT.DB command was requested but failed. The current RDE for non-HALDB full-function database or HALDB partition *dddddddd* or area *aaaaaaaa* of database *dddddddd* was not replaced, because no REPLACE option was specified in the INIT.DB command control statement. If you already 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: *ddddddd***

## Explanation

The ACB or DBD member is not found for the specified database or HALDB partition *ddddddd*.

## 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: *ddddddd* NOT PROCESSED. REASON: *text***

## Explanation

Database *ddddddd* was not processed. The reason is shown in *text*.

***text***  
description

### LOGICAL DBD

DMB verification is not needed for logical DBD.

### MSDB DBD

MSDB DBD is not supported.

### GSAM DBD

GSAM DBD is not supported.

### HALDB WITHOUT RECON

HALDB DBD requires RECONx DD statements or the DFSMDA dynamic allocation members, but they are not specified.

### ISAM ACCESS METHOD

ISAM access method is used. It is not supported.

### INCOMPAT DMB

Incompatible level of ACB member.

### SHR INDEX NOT 1ST

DMB verification is needed only for the first shared index.

### IMS SYSTEM MEMBER

The specified ACB has an IMS system member name, which is not supported.

## System action

The utility skips processing database *ddddddd* 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: *ddddddd* 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  
TIMESTAMP: *tttttttttttt*

## Explanation

No RDE that has time stamp *tttttttttttt* (or CURRENT) 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:  
*dddddddd* [AREA: *aaaaaaaa*]

## Explanation

No RDE was found for non-HALDB full-function database or HALDB partition *dddddddd* or area *aaaaaaaa* of database *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 correct database name is supplied. Correct the error, and rerun the utility job.

---

**FABL0463E** NO RDE FOUND FOR DATABASE:  
*dddddddd* [AREA: *aaaaaaaa*] AND  
TIMESTAMP: *tttttttttttt*

## Explanation

No RDE was found for non-HALDB full-function database or HALDB partition *dddddddd* or area *aaaaaaaa* of database *dbname* and time stamp *tttttttttttt*.

## 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.

---

**FABLO481E 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.

---

**FABLO482E 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.

---

**FABLO483E A MISMATCH TYPE FOUND BETWEEN DBD/ACB *mmmmmmmm* AND RECON**

## Explanation

A mismatch of the database type was found between the DBD or the ACB *mmmmmmmm* 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.

---

**FABLO484E 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.

---

**FABLO485E 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  
ddddddd IS AN INCORRECT  
DATABASE ORGANIZATION**

### Explanation

The specified database *ddddddd* 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 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: yyyy.ddd  
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 *ddddddd* or area *aaaaaaaa* of database *ddddddd* at UTC time stamp *yyyy.ddd 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: yyyy.ddd  
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 *ddddddd* or area *aaaaaaaa* of database *ddddddd* at UTC time stamp *yyyy.ddd 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 (PARAM: *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 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 (IARR2V-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: *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:  
*ddname*

## 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: *ddname*

## 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: *ddname***

## 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 error. Collect the dump, and contact IBM Software Support.

---

**FABL1021E      LIBRARY MISSING: *ddname* DD**

## Explanation

Data set *ddname* could not be opened during initialization. The *ddname* is the DD name of the DBDLIB/PSBLIB/ACBLIB data set.

## System action

IMS Library Integrity Utilities Interface returns return code 16 to the caller.

## User response

Determine whether the required libraries are specified in DD statements. Correct the problem, and rerun the program.

---

**FABL1022E      DYNAMIC [ALLOCATION |  
DEALLOCATION] FAILED FOR  
*ddname*: RETURN CODE=*xxxx*,  
REASON CODE=*yyyy***

## Explanation

An attempt to dynamically allocate or deallocate the *ddname* data set failed. *xxxx* is the hexadecimal return code, and *yyyy* is the hexadecimal reason 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.

---

**FABL1023E      *ddname* DID NOT OPEN**

## Explanation

The *ddname* data set that was allocated dynamically could not be opened during initialization.



## 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 job log and the dump, and contact IBM Software Support.

---

**FABL1024E      ERROR LOADING MODULE *module*:  
ABEND CODE=*nnnn* REASON  
CODE=*mmmm***

## Explanation

An error occurred while load module *module* was being loaded. *nnnn* is the hexadecimal system abend code, and *mmmm* 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 *ddname1* [OR  
*ddname2*]**

## 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. *ddname1* is the DD name of the SYSPRINT data set, and *ddname2* 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*(/*sss*)**

## Explanation

An error occurred in module *module*. *uuuu* is the user completion code, and *sss* 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 *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 *dbname***

### Explanation

The Check function is not done for *dbname*, 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.

### User response

Correct the member name in the control statement, and rerun the job.

---

**FABL2013E      INCORRECT PARAMETER IS SPECIFIED**

### Explanation

An incorrect parameter was found in the control statement of the SYSIN data set.

### System action

The Consistency Checker utility ends with a return code of 12.

### User response

Correct the parameter of the 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 *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.

---

<b>FABL2020E</b>	<b>NO MEMBER FOUND FOR <i>member</i> IN [DBDLIB   PSBLIB]</b>
------------------	---

## 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.

---

<b>FABL2021E</b>	<b><i>member</i> IS NOT A CORRECT [DBD   PSB]: ERROR IS DETECTED IN <i>block</i></b>
------------------	--

## Explanation

DBD or PSB *member* was loaded, but was found not to be a valid DBD or PSB. If the incorrect block can be identified, the block name is shown in the message.

## System action

The Consistency Checker utility continues processing without reporting for this DBD or PSB member.

## User response

Check whether the member is a DBD or a PSB. If the member is a DBD or a PSB, regenerate it.

---

<b>FABL2022E</b>	<b><i>member</i> WAS NOT [DBD   PSB] TYPE ACB MEMBER</b>
------------------	--

## 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.

---

<b>FABL2023E</b>	<b>IMS VERSION INCONSISTENT BETWEEN ACB <i>x.x</i> AND DFSRESLB <i>y.y</i></b>
------------------	--

## 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.

---

<b>FABL2024E</b>	<b>DUPLICATE MEMBER FOUND FOR <i>member</i> IN [DBDLIB   PSBLIB] <i>+nnn</i></b>
------------------	--

## Explanation

Duplicate member was found in DBDLIB or PSBLIB. The *member* is the specified member name. *+nnn* is the relative position of a data set within a concatenation of data sets.

## System action

The Consistency Checker utility continues processing without checking this DBD or PSB member.

## User response

Check the duplicate DBD or PSB member.

---

<b>FABL2025E</b>	<b>RDJFCB MACRO FAILED FOR <i>ddname</i>: RETURN CODE=<i>rrrr</i></b>
------------------	---

## Explanation

The RDJFCB macro attempted for the indicated DD, but failed.

### 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.

---

**FABL2026I**      **THE DATA SET THAT IS SPECIFIED BY NSYSRDDSDD IS A SYSTEM RDDSDD**

### Explanation

The RDDSDD specified on the NSYSRDDSDD is a system RDDSDD.

### System action

Consistency Checker continues processing.

### User response

None. This message is informational.

---

**FABL2030E**      *ddname* **DID NOT OPEN**

### Explanation

The data set *ddname* could not be opened during initialization.

### System action

The Consistency Checker utility ends abnormally.

### User response

Check the cause of this failure.

---

**FABL2031E**      **UNSUPPORTED IMS VERSION: x.x**

### Explanation

DFSRESLB IMS version x.x is not supported.

### System action

The Consistency Checker utility ends with a return code of 12.

### User response

Check the IMS version of DFSRESLB.

---

**FABL2032E**      **INCORRECT [DFSDDIRx | DFSPDIRx] IN MODBLKS**

### Explanation

The online database definition DFSDDIRx, the online application program definition DFSPDIRx is incorrect, or the IMS version used for it is not the same as the one used for DFSRESLB.

### System action

The Consistency Checker utility ends abnormally.

### User response

Check the cause of the error. Correct the problem, and rerun the program.

---

**FABL2033E**      **UNKNOWN RECON LISTING FOUND**

### Explanation

To obtain DBRC information, the Consistency Checker utility linked internally to the DBRC utility DSPURXRT. However, an unknown RECON listing was found.

### System action

The Consistency Checker utility ends abnormally.

### User response

Check the cause of the error. Correct the problem, and rerun the program.

---

**FABL2034E**      **UNSUPPORTED IMS VERSION: x.x for PSB CHECK PROCESS**

### Explanation

DFSRESLB IMS version x.x is not supported for the consistency check of PSBs.

### System action

The Consistency Checker utility ends with a return code of 12.

### User response

Check the IMS version of DFSRESLB.

---

**FABL2035E**      **GETMAIN FAILED**

### Explanation

The program could not obtain enough area with the GETMAIN macro.

## System action

The Consistency Checker 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 program.

---

**FABL2040E**      **ERROR LOADING [DBDLIB  
| PSBLIB] member:  
(ABEND CODE=nnnn REASON  
CODE=mmmm)**

## Explanation

An error has occurred during DBD or PSB *member* load. *nnnn* is the hexadecimal system abend code, and *mmmm* is the hexadecimal reason code.

## System action

The Consistency Checker utility continues processing without reporting for this DBD or PSB member.

## User response

Check the cause of this load error. Correct the problem, and rerun the program.

---

**FABL2041E**      **ERROR LOADING *ddname member*:  
(ABEND CODE=nnnn REASON  
CODE=mmmm)**

## Explanation

An error occurred during *ddname member* load. *nnnn* 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 *ddname member*:  
(ABEND CODE=nnnn REASON  
CODE=mmmm)**

## Explanation

An error occurred during *ddname member* load. *nnnn* 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 *ddname*: RETURN CODE=*xxxx*, REASON CODE=*yyyy***

## Explanation

An attempt to dynamically allocate, deallocate, or concatenate the *ddname* 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**      ***ddname* DID NOT OPEN**

## Explanation

The *ddname* 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 job log and the dump, and contact IBM Software Support.

---

**FABL2052E**      **DBRC COMMAND FAILED: RETURN CODE=*nnnn***

## Explanation

The Consistency Checker utility internally linked DBRC utility DSPURXRT to obtain DBRC information. *nnnn* 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 *ddname1* DD OR *ddname2* 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 *ddname* 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 *ddname* 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 DYNALOC FAILED FOR DDNAME: ddname DSNNAME: dsn RC=rr RSN=ssss**

## Explanation

A dynamic allocation request for DDNAME *ddname* DSNNAME *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 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 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 (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.

---

**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 LIUGINST) 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 *LIUGimsid*) 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 RC=rr RSN=ssss***

### Explanation

The OPEN macro for the VSAM data set *ddname* failed. The return code is *rr*, and the reason code is *ssss*.

### 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 RC=rr***

### 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

Check whether the correct data set is used for the DD name. For the return code of the OPEN macro, see the *z/OS DFSMS Macro Instructions for Data Sets*. 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.

---

**FABL3008E      GETMAIN FAILED FOR SIZE=*size*  
RC=*rr***

## Explanation

The GETMAIN macro for storage (size=*size*) 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

Ensure that the REGION parameter for the JOB or EXEC statement is large enough. Increase the region size, and restart the IMS online subsystem or rerun the 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.

---

**FABL3009E      MODULE NOT FOUND FOR *module*  
IN DDNAME: *ddname***

## Explanation

The *module* module was not found in the *ddname* concatenation.

## System action

The IMS batch job ends abnormally.

## User response

Determine if the module is provided for the *ddname* concatenation of the batch job. Supply the module to the *ddname* concatenation and rerun the batch job.

---

**FABL3010E      THE BPE CONFIGURATION  
MEMBER IS INCORRECT FOR LIU  
INTEGRITY CHECKER**

## Explanation

The BPE configuration parameter member or the user 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

Verify that the EXITMBR statement for the DBRC user exit list member is in the BPE configuration parameter member, and that the FABLBINO module is specified only in the DBRC 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 57. Do not issue the BPE USEREXIT command until you restart the subsystem.

---

**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 FABLBINO 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 IMS PROCLIB, see “Configuring for a BPE-based DBRC environment” on page 57. 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: *nn.n***

## Explanation

The Integrity Checker utility was run on an unsupported version of IMS. *nn.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.

---

<b>FABL3301E</b>	<b>VSAM macro FAILED FOR DDNAME: <i>ddname</i> RC=<i>rr</i> RSN=<i>ssss</i></b>
------------------	---

---

## Explanation

The VSAM macro *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.

---

<b>FABL3302E</b>	<b>VSAM macro FAILED FOR DDNAME: <i>ddname</i> RC=<i>rr</i> RSN=<i>ssss</i></b>
------------------	---

---

## Explanation

The VSAM macro *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.

---

<b>FABL3303E</b>	<b>VSAM SHOWCB FAILED FOR DDNAME: <i>ddname</i> RC=<i>rr</i> RSN=<i>ssss</i></b>
------------------	--

---

## 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.

---

<b>FABL3304E</b>	<b>ENQ FAILED FOR LICON DATA SET. RC=<i>rr</i></b>
------------------	--

---

## Explanation

The ENQ macro for the LICON data set failed. The return code is *rr*.

## 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 LICON data set is used. For the return code, see *z/OS MVS Programming: Assembler Services Reference, Volume 1 (ABEND-HSPSERV)*. Correct the error, and rerun the batch job. If necessary, restart the IMS subsystem.

---

**FABL3305E**      **LICON DATA SET IS IN AN OLDER  
FORMAT**

### Explanation

The LICON data set might not be converted to the latest format.

### System action

The IMS online subsystem that has a BPE-based DBRC region continues processing with the Integrity Checker utility deactivated. The IMS online subsystem that has a non-BPE-based DBRC region, the IMS batch job, or the LICON utility ends abnormally.

### User response

Check whether the LICON data set has been converted to the latest format. If not, convert the LICON data set, and restart IMS online subsystem or rerun the 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.

---

**FABL3400E**      **OPEN FAILED FOR DDNAME:  
*ddname***

### Explanation

The OPEN macro for DD name *ddname* failed.

### System action

The LICON utility job ends abnormally.

### User response

Check whether the correct data set is specified to the DD statement. Correct the error, and rerun the LICON utility job.

---

**FABL3401E**      **CLOSE FAILED FOR DDNAME:  
*ddname***

### Explanation

The CLOSE macro for DD name *ddname* failed.

### System action

The LICON utility job ends abnormally.

### User response

Check whether the correct data set is specified to the DD statement. Correct the error, and rerun the LICON utility job.

---

**FABL3404E**      **MODULE NOT FOUND FOR *module***

### Explanation

Module *module* was not found in the STEPLIB concatenation.

### System action

The LICON utility job ends abnormally.

### User response

Check whether the correct program libraries are concatenated to the STEPLIB DD statement. Correct the error, and rerun the job.

---

**FABL3405E**      **LOAD FAILED FOR MODULE:  
*module***

### Explanation

The LOAD macro for the indicated module has failed.

### System action

The LICON utility job ends abnormally.

### User response

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.

---

**FABL3406E**      **GETMAIN FAILED FOR SIZE=*size*  
RC=*rr***

### Explanation

The GETMAIN macro for storage (*size=size*) failed. The return code is *rr*.

### System action

The LICON 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.

---

**FABL3407E**      **READ ERROR FOR PDS  
DIRECTORY OF [ACBLIB | DBDLIB]**

## Explanation

The READ macro to read the PDS directory of the ACBLIB DD or the DBDLIB DD failed.

## System action

The LICON utility job ends abnormally.

## User response

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.

---

**FABL3408E FIND FAILED FOR ACBLIB. RC=rr**

## Explanation

The FIND macro issued for the ACBLIB DD failed. The return code is *rr*.

## System action

The LICON utility job ends abnormally.

## User response

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.

---

**FABL3409E 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 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.

---

**FABL3410E VSAM MODCB FAILED FOR DDNAME: *ddname* RC=rr RSN=ssss**

## Explanation

The VSAM MODCB 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 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 LICON utility job.

---

**FABL3411E GET FAILED FOR DDNAME: *ddname***

## Explanation

The GET macro failed for the data set whose DD name is *ddname*.

## System action

The LICON utility job ends abnormally.

## User response

Check whether the correct data set is specified. Correct the error, and rerun the LICON utility job.

---

**FABL3412E PUT FAILED FOR DDNAME: *ddname***

## Explanation

The PUT macro failed for the data set whose DD name is *ddname*.

## System action

The LICON utility job ends abnormally.

## User response

Check whether the correct data set is specified. Ensure that enough space is allocated to the data set. Correct the error, and rerun the LICON utility job.

---

**FABL3413E BLDL FOR ACB MEMBER *member* FAILED**

## Explanation

The BLDL macro for the indicated ACB member has failed.

## System action

The LICON utility job ends abnormally.

## 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.

---

**FABL3416E IMS TOOLS BASE V1.6 OR LATER IS REQUIRED**

## Explanation

The requested function requires IBM IMS Tools Base for z/OS 1.6 or later. However, the load module library of IMS Tools Base 1.6 or later 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

Specify the load module library of IMS Tools Base 1.6 or later to the STEPLIB DD concatenation.

---

**FABL3417E UNSUPPORTED LEVEL OF IMS IS BEING USED FOR IMS DIRECTORY: xx.x**

## Explanation

The LICON utility with IMS directory 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 job step return code of 16.

## User response

Check whether the version of IMS is correct. Correct the error, and rerun the utility.

---

**FABL3418E ACCESS TO THE IMS DIRECTORY FAILED WITH RC=rc AND RSN=rsn. FUNCTION=func**

## Explanation

The LICON utility cannot access the IMS catalog. This message is issued when the utility loads ACBs from the IMS directory but fails to access the IMS directory.

## System action

The LICON utility ends immediately with a job step return code of 16.

## User response

Ensure that the following IMS catalog parameter is specified correctly, and then rerun the job:

- Parameter in the FABLIN DD data set.

---

**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=*dbdname*  
DDN=*ddname* KEYTYPE=*keytype* accesstype
- 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***

## 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 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.

## 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

None. This message is informational.

---

**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 PSB 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      SYSOUT DID NOT OPEN**

### Explanation

The SYSOUT 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 *xxxxxxxx***

### Explanation

DBD/PSB *xxxxxxxx* 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 *x.x***

### 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.

---

<b>FABM0043W</b>	<b>REAL LOGICAL CHILD SEGMENT \$SEGMnnn AND DBD NAME \$DBDnnn ON DBD dbdname ARE NAMED BY DBD/PSB/ACB MAPPER AUTOMATICALLY</b>
------------------	--

## 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

## 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. Also, some messages that are issued when you use the Catalog Manager utility or when Library Integrity Utilities is run under IMS Administration Tool also begin with the prefix FABN.

---

<b>FABN0001E</b>	<b>SYSIN DID NOT OPEN</b>
------------------	---------------------------

## Explanation

The SYSIN data set could not be opened during initialization.

## System action

The DBD/PSB/ACB Reversal utility ends abnormally.

## User response

Determine the cause of the open failure. Correct the problem, and rerun the program.

---

<b>FABN0003E</b>	<b>SYSOUT DID NOT OPEN</b>
------------------	----------------------------

## Explanation

The SYSOUT data set could not be opened during initialization.

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.

---

<b>FABM0044W</b>	<b>UNSUPPORTED DBD member IN DBDLIB</b>
------------------	---

## 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.

## System action

The DBD/PSB/ACB Reversal utility ends abnormally.

## User response

Determine the cause of the open failure. Correct the problem, and rerun the program.

---

<b>FABN0004E</b>	<b>FABNRVRS MAXIMUM SAVEAREA COUNT EXCEEDED</b>
------------------	---

## Explanation

The save area count exceeds the maximum count for this utility.

## System action

The DBD/PSB/ACB Reversal utility ends abnormally.

### User response

Determine the cause of the error. Correct the problem, and rerun the program.

---

**FABN0005E      FABNRVRS NO MORE VIRTUAL  
STORAGE AVAILABLE**

### Explanation

The program could not obtain sufficient area with the GETMAIN macro.

### System action

The DBD/PSB/ACB Reversal utility ends abnormally.

### User response

Increase the region size in the JOB statement in the JCL, and rerun the utility.

---

**FABN0006I      BLANK RECORD IN SYSIN**

### Explanation

There is a blank record in the SYSIN data set.

### System action

The DBD/PSB/ACB Reversal utility continues processing.

### User response

None. This message is informational.

---

**FABN0007E      UNKNOWN KEYWORD IN SYSIN**

### Explanation

An unknown keyword is found in the SYSIN data set.

### System action

The DBD/PSB/ACB Reversal utility ends with a return code of 16.

### User response

Correct the keyword, and rerun the utility.

---

**FABN0008W      UNKNOWN OPERAND IN SYSIN**

### Explanation

There is an unknown operand in the SYSIN data set.

### System action

The DBD/PSB/ACB Reversal utility skips this record and continues processing.

### User response

Correct the operand, and rerun the utility.

---

**FABN0009W      UNKNOWN OPTION IN SYSIN**

### Explanation

There is an unknown option in the SYSIN data set.

### System action

The DBD/PSB/ACB Reversal utility skips this record and continues processing.

### User response

Correct the option, and rerun the utility.

---

**FABN0010W      INVALID RECORD IN SYSIN**

### Explanation

There is a record with an invalid format in the SYSIN data set.

### System action

The DBD/PSB/ACB Reversal utility skips this record and continues processing.

### User response

Correct the SYSIN record, and rerun the utility.

---

**FABN0011W      NO DATA IN SYSIN**

### Explanation

There is no valid record in the SYSIN data set.

### System action

The DBD/PSB/ACB Reversal utility ends without producing reports or source codes.

### User response

Specify valid control records in the SYSIN data set, and rerun the utility.

---

**FABN0012E      DBD NAMED [*dbdname* | *acbname*]  
HAS UNKNOWN ACCESS CODE -  
HEX *nn***



## Explanation

The loaded DBD (or DBD-type ACB) *dbdname* has an unknown access method code. *nn* is the unknown code, in hexadecimal.

## System action

Ends the reversal process for this DBD and issues an error message with dump of this module to the SYSOUT data set. Then continues processing with the next module.

## 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 loading 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=*mmmm*)

## Explanation

An error occurred during DBD/PSB member load. *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 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 member name or 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'*nn*'  
FOR FIELD *fieldname* IN SEGMENT  
*segname* IN DBD [*dbdname* |  
*acbname*]

## Explanation

The DBD has an unknown type option in the FIELD statement. X'*nn*' is the unknown type option in hexadecimal, *fieldname* is the FIELD name, *segname* is the SEGMENT name, and *dbdname* 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. *segment* is the name of the segment that has an incorrect SEQ-type option, and *dbdname* 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 (*xxxxxxx*) 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. *xxxxxxx* is the DBD/PSB with a mismatched member name, and *yyyyyyyy* 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 xxxxxxxx**

### Explanation

DBD/PSB xxxxxxxx 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=mmmm)**

## 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 |  
                  IMS] DIRECTORY**

## Explanation

An error occurred while a BLDL macro was being issued for the ACB library, PSB library, or the IMS directory.

## System action

IMS Library Integrity Utilities 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 \$FABN*nnn*  
                  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. \$FABN*nnn* is the name assigned, *nnn* 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=x.x**

## 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 x.x, ACB  
member2 IS IMS y.y**

## Explanation

PSB-type ACB or DBD-type ACB (*member1*) is generated by IMS version x.x, but the DBD-type ACB (*member2*) referred to by the PSB-type ACB or DBD-type ACB (*member1*) 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 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 = *nnnn* 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. *nnnn* 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*. Correct 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*nnnn* 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. \$DBD*nnnn* is the name assigned; *nnnn* 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. \$FABN*nnn* 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 skips this control statement and continues processing.

### User response

None.

---

**FABN0058W NO PCBNAM PREFIX IS SPECIFIED**

### Explanation

Prefixes for PCB names are not specified for the PCBNAMEX option.



## System action

The DBD/PSB/ACB Reversal utility skips this control statement and 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 skips this control statement and 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 proceeds with the specified PCBNAMEX option. If one or more PCBs that are not named in the PSB are found, the DBD/PSB/ACB Reversal utility gives the names to the PCBs by using the specified prefix *prfx*.

## System action

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: xxxxxxxxxxxx**

## 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=*xxxxxxx* 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 xxxLIB**

## 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 \$SEGMnnn AND DBD NAME \$DBDnnn ON DBD *dbdname* ARE NAMED BY DBD/PSB/ACB REVERSAL AUTOMATICALLY**

## Explanation

Real logical child segment name \$SEGMnnn 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 RDMVTAB 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 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 Reversal utility.

Detectable DBD type	Field	Description
DBD and DBD TYPE ACB	DMBDASZE	The size of RDMVTAB CSECT
DBD and DBD TYPE ACB	DMBDAKL	The executable key length of root
DBD	DMBDANME	The name of randomizer module
DBD	DMBDARAP	The number of root anchor points or blocks
DBD	DMBDABLK	The number of the highest blocks that are directly addressed
DBD	DMBDABYM	The maximum number of bytes

## 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 FABNCTLO IS USED**

## Explanation

The site default table module FABNCTLO 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 FABNCTLO IS CORRUPTED**

## Explanation

The site default table module (FABNCTLO) 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 FABNCTLO module and ensure that the first 8 bytes of the site default table shows FABNCTLO. 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. [EXIT PARAMETERS OF DBD AND SEGM STATEMENTS ARE 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 231 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 *IMS Tools Base: IMS Tools Common Services User's Guide and Reference*. 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.

---

**FABN0087W      SENSEG STATEMENT FOR DB PCB OF MSDB IS NOT DECODED.****Explanation**

The parameter is not decoded because this parameter could not be obtained from the MSDB member.

**System action**

IMS Library Integrity Utilities continues processing.

**User response**

None.

---

**FABN0088I      VERSION PARAMETER OF DBD STATEMENT IS DECODED FROM THE ACTIVE ACB DATA SETS OF THE IMS DIRECTORY. [EXIT PARAMETERS OF DBD AND SEGM STATEMENTS ARE DECODED FROM THE ACTIVE ACB DATA SETS OF THE IMS DIRECTORY.]****Explanation**

The following parameters were decoded from an active ACB in the IMS directory.

- The VERSION parameter of the DBD statement
- The EXIT parameter of the DBD and SEGM statements

This happens when the utility tries to decode DBDs in the IMS directory staging data set and the PSB that refers to the DBD does not exist in the IMS directory staging data set.

When the utility decodes a DBD and it detects missing parameters, it looks for the PSB that refers to the DBD to supplement the missing parameters. Because a staging data set does not store all the PSBs – it stores modified PSBs only – if the utility cannot find the relevant PSB in the staging data set, it looks for the PSB in the active ACB data sets of the IMS directory and uses the information in the active ACB to supplement the missing parameters.

**Note:** VERSION and EXIT parameters are supplemented only if the organization of the database is DEDB or MSDB.

**System action**

The Catalog Manager utility continues processing.

**User response**

None. This message is informational.

---

**FABN0089I      SENSEG STATEMENT IS DECODED FROM THE ACTIVE ACB DATA SETS OF THE IMS DIRECTORY.****Explanation**

The SENSEG statement is decoded from an active ACB in the IMS directory. This happens when the utility tries to decode DBDs in the IMS directory staging data set and the PSB that refers to the DBD does not exist in the IMS directory staging data set.

When the utility decodes a PSB and it detects a missing SENSEG statement, it looks for the DBD that the PSB refers to to supplement the missing statement. Because a staging data set does not store all the DBDs – it stores modified DBDs only – if the utility cannot find the relevant DBD in the staging data set, it looks for the DBD in the active ACB data sets of the IMS directory and uses the information in the active ACB to supplement the missing SENSEG statement.

**Note:** The SENSEG statement is supplemented only when the PSB has a DB PCB for an MSDB.

**System action**

The Catalog Manager utility continues processing.

**User response**

None. This message is informational.

---

**FABN0090W      VENDOR SECTION EXISTS.****Explanation**

While processing DBD or PSB, the DBD/PSB/ACB Reversal utility detected a vendor section. This section is not decoded.

**System action**

The DBD/PSB/ACB Reversal utility continues processing.

**User response**

Add the vendor section as needed when you regenerate DBD or PSB.

---

**FABN0091E      *member1* REFERRED BY *member2* NOT FOUND IN IMS DIRECTORY.****Explanation**

An error occurred while reading *member1* which is referred to by *member2*.

When the Catalog Manager utility processes a DB PCB in a PSB and the DB PCB is for MSDB, the utility refers to the MSDB to obtain the value of the SENSEG statement. When the utility processes a PSB in the IMS directory active data set, the MSDB being referred to should also be in the IMS directory active data set. This message is issued when the utility cannot find the MSDB.

### System action

IMS Library Integrity Utilities skips reading *member1*.

### User response

Determine the cause of the error, correct it and rerun the utility.

---

**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.

### User response

None. This message is informational.

---

**FABN1023I      *keyword* OPTION USED: *option***

## Explanation

The site default generation utility proceeds with the runtime option.

## System action

The site default generation utility continues processing.

## User response

None. This message is informational.

---

**FABN1030I      SITE DEFAULT TABLE FABNCTLO IS PRINTED**

## Explanation

The specification in the site default table is reported by the FABNTGEN program.

## System action

The site default generation utility continues processing.

## User response

None. This message is informational.

---

**FABN1100W      FABNTGEN ENDED WITH WARNINGS**

## Explanation

The site default generation utility ended with warning conditions.

## System action

The site default generation utility ends with a return code of 4.

## User response

Check the preceding messages that explain the warning conditions. If necessary, correct the warning conditions and rerun the utility.

---

**FABN1107E      UNKNOWN KEYWORD IN SYSIN:  
keyword**

## Explanation

*keyword*, which is specified in the SYSIN data set, is not a valid keyword.

## System action

The site default generation utility ends with a return code of 16.

## User response

Determine if the correct input control statement for site default is supplied. If necessary, correct the keyword and rerun the utility.

---

**FABN1109E      UNKNOWN OPTION IN SYSIN:  
option**

## Explanation

*option*, which is specified in the SYSIN data set, is not a valid option.

## System action

The site default generation utility ends with a return code of 16.

## User response

Determine if the correct input control statement for site default is supplied. If necessary, correct the option and rerun the utility.

---

**FABN1110W      INVALID RECORD IN SYSIN**

## Explanation

A record in an invalid format is in the SYSIN data set.

## System action

The site default generation utility skips this record and continues processing.

## User response

Determine if the correct input control statement for site default is supplied. If necessary, correct the record and rerun the utility.

---

**FABN1111E      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 site default generation 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.

---

**FABN1200E      FABNTGEN ENDED WITH ERRORS**

## Explanation

The site default generation utility ended with errors.

## System action

The site default generation utility ends with a return code of 8.

## User response

Check the preceding messages that explain the error conditions. Correct the error conditions and rerun the utility.

---

**FABN1201E      SITE DEFAULT TABLE FABNCTLO IS CORRUPTED**

## Explanation

The site default table module (FABNCTLO) is corrupted and site default values are not reported.

## System action

The site default generation utility ends with a return code of 8.

## User response

Specify the correct FABNCTLO module and ensure that the first 8 bytes of the site default table shows FABNCTLO. If the module is damaged, re-create another site default table module and store it in the STEPLIB data set.

---

**FABN1203E      SITE DEFAULT TABLE FABNCTLO IS NOT FOUND**

## Explanation

The site default values were not reported because the site default table module (FABNCTLO) is not in the STEPLIB data sets.

## System action

The site default generation utility ends with a return code of 8.

## User response

Specify the data set that includes the site default table module (FABNCTLO) 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.

## FABQ messages

Messages that are issued by the Advanced ACBGEN utility begin with the prefix FABQ.

---

**FABQ0001E      UNABLE TO OPEN SYSPRINT --  
JOB TERMINATED**

## Explanation

Either the SYSPRINT DD statement is missing or OPEN processing failed for the data set that was specified.

## System action

Processing ends with an abend code of 949.

## User response

Ensure that the SYSPRINT DD exists, and that it specifies valid DD parameters. Correct any errors, and rerun the job.

---

**FABQ0002E      UNABLE TO OPEN  
DDNAME=*ddname***

## 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.

---

**FABQ0003E      DDNAME=*ddname* NOT FOUND IN  
JCL**

## Explanation

The data set in the specified DD JCL statement could not be opened.

## System action

Processing ends with a return code of 16.

## User response

Ensure that the specified DD statement contains valid DD parameters. Correct any errors, and rerun the job.

## System action

Processing ends with a return code of 16.

## User response

Ensure that the specified DD statement contains valid DD parameters. Correct any errors, and rerun the job.

---

**FABQ0003E      DDNAME=*ddname* NOT FOUND IN  
JCL**

## Explanation

The specified DD statement could not be found in the JCL of the job step.

## System action

Processing ends with a return code of 16.

## User response

Ensure that the specified DD statement exists, and rerun the job.

---

**FABQ0004W**      **DDNAME=ddname DATASET  
EMPTY**

## Explanation

This message is an informational message.

## System action

Processing continues, but the return code is set to 4.

## User response

None.

---

**FABQ0005I**      **OTHER REPORTS ARE PRINTED IN  
THE FABQRPT DATA SET BECAUSE  
GENDATE=YES**

## Explanation

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.

## System action

Processing continues.

## User response

View other reports in the FABQRPT data set. If you did not specify the FABQRPT DD statement, the reports are printed to SYSOUT.

---

**FABQ0006E**      **DYNALLO MACRO FAILED  
TO ALLOCATE ddname RC=rr  
RSN=ssss**

## Explanation

The *ddname* data set was not dynamically allocated. *rr* is the hexadecimal return code, and *ssss* is the hexadecimal reason code of the DYNALLO macro.

If this message is issued for the FABQRPT data set, the message implies that the dynamic allocation took place because GENDATE=YES is specified in the LISTLIB command in the ACBSYSIN control statement, but the FABQRPT DD statement is not specified.

## System action

Processing ends with a return code of 16.

## User response

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.

---

**FABQ0007E**      **DUMMY SPECIFIED FOR THE  
ddname DD STATEMENT**

## Explanation

The reports are not generated because DUMMY is specified for the indicated DD statement.

## System action

Processing ends with a return code of 16.

## User response

Correct the DD statement so that it does not specify DUMMY, and then rerun the job.

---

**FABQ0104E**      **UNABLE TO LOAD NAME=module**

## Explanation

An unsuccessful attempt was made to load the specified module. Standard MVS facilities were used to attempt the loading of the specified module.

## System action

Processing ends with a return code of 16.

## User response

Verify that the library containing the module is included in the STEPLIB DD JCL or in your MVS system's link list.

---

**FABQ0108E**      **IMS 'x.x.x' NOT SUPPORTED**

## Explanation

This version/release of IMS is not supported.

## System action

Processing ends with a return code of 16.

## User response

Ensure that your DFSRESLB DD statement contains the correct IMS load module library. 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, DFSUAMBO, DFSDLBLO, 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.

---

**FABQ0200I**      **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 FABQ02nn 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 357](#) 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 357 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 357 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 357 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 357 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 357 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 357 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 357 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 357 for a description of the ACBSYSIN syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

**FABQ0224E**      **'keyword' KEYWORD REQUIRES PARENS AROUND LIST OF SUBPARMS**

## Explanation

A control statement in the ACBSYSIN data set contains a keyword operand name whose data value is more than one subparameter. Multiple subparameters for a given keyword operand must be enclosed in parenthesis.

## System action

Processing ends with a return code of 16.

## User response

See “ACBSYSIN control statements” on page 357 for a description of the syntax rules. Correct the error,

and rerun the job. If the error persists, contact IBM Software Support.

---

**FABQ0225E**      **'keyword' KEYWORD HAS INVALID CONTINUATION CHARACTER**

## Explanation

A control statement in the ACBSYSIN data set contains a keyword operand whose data value is not terminated with either a blank or comma.

## System action

Processing ends with a return code of 16.

## User response

See “Syntax rules” on page 357 for a description of the ACBSYSIN syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

**FABQ0226E**      **'keyword' KEYWORD IS REQUIRED**

## Explanation

A control statement in the ACBSYSIN data set does not contain a required keyword operand.

## System action

Processing ends with a return code of 16.

## User response

See “ACBSYSIN control statements” on page 357 for a description of the syntax rules. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

**FABQ0229E**      **KEYWORD OPERAND EXPECTED -- NONE FOUND**

## Explanation

A control statement in the ACBSYSIN data set showed that another keyword operand was present, but none was found.

## System action

Processing ends with a return code of 16.

## User response

See “Syntax rules” on page 357 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 357 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.

---

**FABQ0234E**     **'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 357 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 357 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.

---

**FABQ1000I**     **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.

---

**FABQ1001E**     **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.

---

**FABQ1002E      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.

---

**FABQ1004E      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.

---

**FABQ1005E      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.

---

**FABQ1006E      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.

---

**FABQ1007E      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.

---

**FABQ1008E      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.

---

**FABQ1009E      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.

---

**FABQ1010E      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.

---

**FABQ1011E      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.

---

**FABQ1012E      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.

---

**FABQ1013E      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  
DFSDLBLO**

### 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 DFSUMGTO 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 DFSUMGTO module. If the IMS ACBGEN utility does not end abnormally, 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 end abnormally, 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 DFSRESLB 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 DFSDLBLO

## Explanation

FABQHUK2 was unable to *hook* the DFSDLBLO 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 ENCOUNTERED 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 ENCOUNTERED 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 ENCOUNTERED 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.

---

**FABQ9990I ACBGEN SUBTASK UNABLE TO LOAD ITS OWN COPY OF DFSDLBLO**

---

**Explanation**

This message is issued whenever the MAXTASKS= operand specifies a value greater than 1 and the DFSDLBLO module in the IMS RESLIB does not have the nonreusable link-edit attribute.

**System action**

The subtask that issued this MVS console message ends, but processing continues.

**User response**

If MVS subtasking is wanted, the DFSDLBLO module in the IMS RESLIB must be link-edited as nonreusable.

---

**FABQ9991W An ACBGEN SUBTASK ABENDED WITH SYSTEM CODE OF "xxx" FOR PSB=*psbname***

---

**Explanation**

This message is issued whenever a subtask ends abnormally. 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      nnnnnnn PSBS TO BE PROCESSED  
                  BY ACBGEN**

## Explanation

This message is issued when the MONITOR=(PROGRESS=(Y,*frequency\_value*)) operand is specified in the ACBSYSIN data set. *nnnnnnn* 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=*dbdname*,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 *nnnnnnn*.

## System action

This message is issued to the MVS console when the ACBGEN process begins.

## User response

None. This message is informational.

---

**FABQ9998I      mmmmmmm OF nnnnnnn PSBS  
                  PROCESSED BY ACBGEN**

## Explanation

This message is issued when the MONITOR=(PROGRESS=(Y,*frequency\_value*)) operand is specified in the ACBSYSIN data set. *mmmmmmm* is the number of the members that were just processed, and *nnnnnnn* is the total number to be processed.

If BUILD DBD=*dbdname*,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 *nnnnnnn*.

## System action

This message is issued to the MVS console during the ACBGEN process.

## User response

None. This message is informational.

---

**FABQ9999I      nnnnnnn PSBS PROCESSED BY  
                  ACBGEN**

## Explanation

This message is issued when the MONITOR=(PROGRESS=(Y,*frequency\_value*)) operand is specified in the ACBSYSIN data set. *nnnnnnn* is the number that was actually processed.

## 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.

---

**FABV000E      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

---

**FABV008E**      **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

---

**FABV009E**      **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 FABVDVCT has all the definitions of 3270-Ax listed. Try running the utility again with the correct Device Characteristics Table.

## Module

FABVTRBB

---

<b>FABV0022W</b>	<b>USING DEFAULT DEVICE CHARACTERISTICS TABLE FABVDVCT</b>
------------------	--

## 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

---

<b>FABV0030W</b>	<b>MID/DIF TIMESTAMPS NOT THE SAME FOR MBR <i>member</i></b>
------------------	--

## 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

FABVBLDC

---

<b>FABV0031W</b>	<b>MOD/DOF TIMESTAMPS NOT THE SAME FOR MBR <i>member</i></b>
------------------	--

## 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

FABVBLDC

---

<b>FABV0044I</b>	<b>SOURCE FOR MEMBER <i>member</i> BUILT.</b>
------------------	---

## 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

FABVBLDC

---

**FABV0045E      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

---

**FABV0046E      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 MIDs and MODs on the SELECT statement. Also check that the format library is correctly specified.

### Module

FABVANL

---

**FABV0047E      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

---

<b>FABV0062W</b>	<b>COPY PROCESS COMPLETED, BUT SOME WARNINGS WERE DETECTED</b>
------------------	--

## 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

---

<b>FABV0063E</b>	<b>COPY PROCESS COMPLETED, BUT SOME ERRORS WERE DETECTED</b>
------------------	--

## 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

---

<b>FABV0064E</b>	<b>COPY PROCESS ENDED BECAUSE ERRORS WERE DETECTED</b>
------------------	--

## 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

---

<b>FABV0065E</b>	<b>COPY PROCESS ENDED BECAUSE COPYFMT LIBRARY CONTAINED SOME ALIAS MEMBERS</b>
------------------	--

## 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

---

**FABV0072I**      **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

---

**FABV0073W**      **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

---

**FABV0074W**      **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

---

**FABV0080E**      **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

---

<b>FABV0081E</b>	<b>READ ERROR WHILE COPYING MBR=<i>member</i> IN DD=FORMAT</b>
------------------	--

## 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

---

<b>FABV0082E</b>	<b>WRITE ERROR WHILE COPYING MBR=<i>member</i> TO DD=COPYFMT</b>
------------------	--

## 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

---

<b>FABV0083E</b>	<b>STOW ERROR WHILE COPYING MBR=<i>member</i> TO DD=COPYFMT RSN=<i>reason_code</i> RC=<i>return_code</i> ID=<i>id</i></b>
------------------	---

## 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

---

<b>FABV0084E</b>	<b>READ ERROR WHILE COPYING IN DD=COPYFMT</b>
------------------	---

## 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

FABVCPY

---

**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.

## 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**

## 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=*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

## 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 FABWRRPT data set.

## System action

The Multiple Resource Checker utility continues processing.

## User response

None. This message is informational.

---

<b>FABW0011W</b>	<b>DB TYPES ARE INCONSISTENT. MEMBER <i>member</i> IS NOT COMPARED</b>
------------------	--

## 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 FABWRRPT data set when the utility compares RECON data sets.

## System action

The Multiple Resource Checker utility continues processing.

## User response

None.

---

<b>FABW0012W</b>	<b>ACB <i>member</i> IN ACBLIB<i>xx</i> IS NOT COMPARED BECAUSE IT WAS GENERATED BY AN UNSUPPORTED IMS VERSION</b>
------------------	--

## 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.

---

<b>FABW0013W</b>	<b>RECON<i>xxxn</i> DD STATEMENT MISSING</b>
------------------	--

## 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.

---

<b>FABW1001E</b>	<b>DUPLICATE CONTROL STATEMENT FOUND</b>
------------------	--

## 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.

---

<b>FABW1002E</b>	<b>INCORRECT PARAMETER FOUND ON THE CONTROL STATEMENT</b>
------------------	---

## 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.

---

<b>FABW1003E</b>	<b>INCORRECT MEMBER NAME IS SPECIFIED ON THE CONTROL STATEMENT</b>
------------------	--

## 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 RECONxxxn 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 *ddname* 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 job 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:  
ddname**

## 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) |

## 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 or when Library Integrity Utilities is run under IMS Administration Tool 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

## 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.

---

**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).

---

**FABX0007W      ONE OR MORE INVALID DBD OR PSB RESOURCES FOUND DURING THE 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.

---

**FABX0008W      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.

---

**FABX0009W      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.

---

<b>FABX0010I</b>	<b>THE CATALOG [VALIDATION   COMPARE] PROCESS IS IN PROGRESS. xxx.x % COMPLETE.</b>
------------------	---

## 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.

---

<b>FABX0011W</b>	<b>A WARNING MESSAGE IS PRINTED IN THE ([DBD   PSB] resource) SOURCE.</b>
------------------	---

## 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.

---

<b>FABX0012W</b>	<b>EXPORT ENDED WITH WARNINGS.</b>
------------------	------------------------------------

## Explanation

IMS Library Integrity Utilities detected warning conditions in export processing.

## System action

IMS Library Integrity Utilities ends with a return code of 2.

## 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.

---

<b>FABX0013I</b>	<b>THE DECODED SOURCE CONTAINS STATEMENTS THAT WERE SUPPLEMENTED FROM THE ACTIVE ACB DATA SETS OF THE IMS DIRECTORY.</b>
------------------	--

## Explanation

The following parameters and statements were decoded from active ACBs in the IMS directory:

- The VERSION parameter of the DBD statement
- The EXIT parameter of the DBD and SEGM statements
- The SENSEG statement

This happens when the utility tries to decode a DBD or PSB in the IMS directory staging data set and the PSB that refers to the DBD or the DBD that the PSB refers to does not exist in the IMS directory staging data set.

When the utility decodes a DBD or PSB and it detects missing parameters, it looks for the PSB that refers to the DBD or the DBD that the PSB refers to to supplement the missing parameters. Because a staging data set does not store all the DBDs and PSBs – it stores modified DBDs and PSBs only – if the utility cannot find the relevant DBD or PSB in the staging data set, it looks for the DBD or PSB in the active ACB data sets of the IMS directory and uses the information in the active ACB to supplement the missing parameters.

## System action

The Catalog Manager utility continues processing.

## User response

None. This message is informational.

---

<b>FABX0014W</b>	<b>THE PROGRAM VIEW ENDED WITH WARNINGS BECAUSE THE</b>
------------------	---

**REFERENCED DBD WAS NOT FOUND.**

**Explanation**

DB Schema (DB PCB) list was not generated because the DBD which was referenced by the specified PSB was not found.

**System action**

IMS Library Integrity Utilities ends with a return code of 4.

**User response**

None.

---

**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 parameter" in the *z/OS MVS JCL Reference*.

---

**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 that is shown in the message is the return code from the OPEN macro. Use the information about the return



codes in the topic "OPEN return 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 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 the *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.

### System action

The Catalog Manager utility ends with a return code 8.

### User response

Ensure that the format of the data sets is correct and that the data sets are not damaged. The return code and reason code in the message are from the DESERV 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.

---

**FABX0509E**      **ddname DD IS MISSING.**

### Explanation

The indicated DD is not allocated.

### System action

The Catalog Manager utility ends with a return code 8.

### User response

See "Control statements for the Catalog Manager utility" on page 292 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 DBD OR PSB RESOURCES IN IMS DIRECTORY *ims\_directory\_data\_set*.**

### Explanation

This message indicates that the IMS directory data sets are empty.

*ims\_directory\_data\_set* shows either of the following texts:

#### ACTIVE

The IMS directory data sets for storing active resources are empty.

#### STAGING

The IMS directory staging data set is empty.

### System action

IMS Library Integrity Utilities ends with a return code of 8.

### 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 *ddname* 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 292.

---

**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 292 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 292 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 292 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 292 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.

---

<b>FABX0526E</b>	<b>THE NUMBER OF PARAMETERS SPECIFIED IN <i>keyword</i> KEYWORD EXCEEDED THE LIMIT. MAX IS <i>num</i>.</b>
------------------	--

## 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.

---

<b>FABX0527E</b>	<b>THE <i>n</i>TH PARAMETER ON THE <i>keyword</i> KEYWORD HAS INCORRECT LENGTH.</b>
------------------	---

## 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 292 and correct the length of the indicated parameter. Then, rerun the job.

---

<b>FABX0528E</b>	<b>THE PARAMETER FOR <i>keyword</i> MUST CONSIST ONLY OF ALPHANUMERIC CHARACTERS, \$, #, @, %, AND *.</b>
------------------	---

## Explanation

The parameter specified for the indicated keyword contains one or more unsupported characters.

## System action

The Catalog Manager utility ends with a return code of 8.

## User response

Correct the parameter and rerun the job.

---

<b>FABX0531E</b>	<b>MEMBER <i>member</i> WAS GENERATED BY IMS <i>version</i> OR EARLIER.</b>
------------------	---

## 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 Populate utility (DFS3UACB) that is provided by a supported version of IMS.

---

<b>FABX0532E</b>	<b>THE RESOURCE <i>resource</i> IN THE [ACBLIB   IMS CATALOG   IMS DIRECTORY] IS NOT [DBD   PSB].</b>
------------------	---

## 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.

---

<b>FABX0533W</b>	<b>THE SPECIFIED [DBD   PSB] RESOURCE <i>resource</i> DOES NOT EXIST IN THE [ACBLIB   IMS CATALOG   IMS DIRECTORY].</b>
------------------	---

## Explanation

The specified DBD or PSB resource was not found in the ACBLIB, IMS catalog, or IMS directory.

## System action

IMS Library Integrity Utilities ends with a return code of 2 or 4.

## User response

Specify the correct resource name.

---

**FABX0535E      ACCESS TO THE IMS DIRECTORY  
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 FABXPPRM 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 DFSDFxxx 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 DIRECTORY  
IS NOT DEFINED TO THE IMS.  
[ACBLIB | DBDLIB] IS NOT  
DEFINED TO THE IMS.**

## Explanation

The requested function requires the IMS catalog, IMS directory, DBD library, 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  
FUNC KEYWORD PARAMETERS**

## Explanation

The requested IMS Library Integrity Utilities function requires IBM IMS Tools Base for z/OS 1.6 or later, however the load module library of IMS Tools Base 1.6 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 IMS Tools Base 1.6 or later to the STEPLIB DD concatenation.

If you received this message while using the ISPF interface of IMS Administration Tool, add the IMS Tools Base SGLXLOAD library 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 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 to 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.

---

**FABX0544E**      **IMS CATALOG ANALYSIS CANNOT  
BE PERFORMED FOR NON DATA  
SHARING IMS SYSTEMS.**

## Explanation

IMS catalog analysis cannot be performed because a data sharing group is not defined to the Administration Tool.

IMS catalog analysis issues DL/I calls to the IMS catalog database. Therefore, to run IMS catalog analysis, data sharing must be configured for the IMS systems so that they can communicate with the LPAR where the IMS Tools Base Distributed Access Infrastructure (DAI) server is running.

## System action

IMS Library Integrity Utilities ends with a return code of 99.

## User response

Use the IRLM to configure data sharing for the IMS systems. Then create an IMS data sharing group of IMS Administration Tool and register the IMS systems to the group. The IRLM of one of the IMS systems in the group must be defined to the LPAR where the DAI server is running.

---

**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: *dsname***

### Explanation

IMS catalog database data sets must be OSAM data sets or VSAM linear data sets (OSAM LDSs). 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 *dsname* 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 *dsname* DATA SET FAILED WITH RC=*rc* AND DSCB=*dsch*.**

### 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 *dsname* 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.

---

**FABX0550W**      **THERE ARE NO DBD OR PSB RESOURCES IN IMS DIRECTORY *ims\_directory\_data\_set*.**

### Explanation

IMS Library Integrity Utilities did not export any DBD or PSB resources because the indicated IMS directory data set is empty.

*ims\_directory\_data\_set* shows either of the following texts:

#### ACTIVE

The IMS directory data sets for storing active resources are empty.

#### STAGING

The IMS directory staging data set is empty.

### System action

IMS Library Integrity Utilities ends with a return code of 2.

### User response

None.

---

**FABX0551E**      **THERE ARE NO DBD OR PSB RESOURCES IN THE ACB LIBRARY.**

### Explanation

The ACB library data set is empty.

### System action

IMS Library Integrity Utilities ends with a return code of 8.

## User response

Run the Application Control Block Maintenance utility (DFSUACB0) to store ACBs in the ACB library.

---

**FABX0552E      IMS CATALOG ANALYSIS CANNOT BE PERFORMED FOR IMS SYSTEMS RUNNING IN A DIFFERENT LPAR.**

## Explanation

IMS catalog analysis cannot be performed for an IMS control region that is active in an LPAR where the IMS Tools Base Distributed Access Infrastructure (DAI) server is not running. To perform IMS catalog analysis, the PTF for APAR PI90085 must be applied to IMS Administration Tool.

## System action

IMS Library Integrity Utilities ends with a return code of 99.

## User response

Complete the following steps:

1. Apply the PTF for APAR PI90085 to IMS Administration Tool.
2. Use the IRLM to configure data sharing for the IMS systems.
3. Create an IMS data sharing group of IMS Administration Tool and register the IMS systems to the group. The IRLM of one of the IMS systems in the group must be defined to the LPAR where the DAI server is running.

---

**FABX0553E      SECOND MEMBER NAME CANNOT BE SPECIFIED WHEN COMPARING MEMBERS WITH DBDLIB OR PSBLIB.**

## Explanation

When comparing ACBs in the IMS directory with DBDs or PSBs in DBD or PSB libraries, the Catalog Manager utility does not accept a second member name in the FABXCIN control statement.

## System action

The Catalog Manager utility ends with a return code of 8.

## User response

If NAME1 and NAME2 keywords specify different member names, delete the NAME2 keyword or specify

the same member name for both NAME1 and NAME2 keywords.

---

**FABX0554E      IMS DIRECTORIES MUST BE SPECIFIED WITH THE SAME IMSID.**

## Explanation

IMS Library Integrity Utilities is being used through IMS Administration Tool. IMS Library Integrity Utilities ended with an error because the input parameter provided is incorrect.

## System action

IMS Library Integrity Utilities ends with a return code of 8.

## User response

None.

---

**FABX0555E      MAINTENANCE LEVEL TOO LOW. APPLY MAINTENANCE TO IMS LIBRARY INTEGRITY UTILITIES TO USE THE IMS ADMINISTRATION TOOL CLIENT.**

## Explanation

The maintenance level of IMS Library Integrity Utilities is too low to use the IMS Administration Tool client.

## System action

IMS Library Integrity Utilities ends with a return code of 99.

## User response

Apply maintenance to IMS Library Integrity Utilities to use the IMS Administration Tool client.

---

**FABX0556E      WILD CARD CHARACTERS CANNOT BE SPECIFIED FOR NAME1= WHEN NAME2= IS SPECIFIED.**

## Explanation

Wildcard characters cannot be used to describe NAME1= when NAME2= is specified. You can use wildcard characters for NAME1= if NAME2= is not specified.



### System action

The Catalog Manager utility ends with a return code of 8.

### User response

Correct the keyword parameter and rerun the job.

---

**FABX0557W**      **DIFFERENCE FOUND DURING  
COMPARE ACB=*members***

### Explanation

The compare function ran normally, and a difference was found between the 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 Catalog Manager utility generates a compare report and continues processing.

### User response

None.

---

**FABX0558E**      ***member* IS AN INTERNAL MEMBER  
USED BY IMS. THE UTILITY  
CANNOT PROCESS THIS MEMBER.**

### Explanation

The indicated member cannot be processed because it is an internal member used by IMS and it is not a DBD or PSB member.

### System action

The Catalog Manager utility ends with a return code of 8.

### User response

Remove this member from the control statements and rerun the job.

---

**FABX0559W**      **THE SPECIFIED [DBD|PSB]  
INSTANCE DOES NOT EXIST IN  
THE IMS CATALOG.**

### Explanation

The specified DBD or PSB instance was not found in the IMS catalog database.

### System action

IMS Library Integrity Utilities ends with a return code of 4.

### User response

Specify the correct resource name.

---

**FABX0560W**      **VERSION PARAMETER OF THE  
DBD STATEMENT IS NOT  
DECODED.**

### Explanation

The VERSION parameter of the DBD statement is not decoded because this parameter could not be obtained from the DBD member.

### System action

IMS Library Integrity Utilities continues processing.

### User response

None.

---

**FABX0561W**      **VENDOR SECTION EXISTS.**

### Explanation

While processing DBD or PSB, IMS Library Integrity Utilities detected a vendor section. This section is not decoded.

### System action

IMS Library Integrity Utilities continues processing.

### User response

Add the vendor section as needed when you regenerate DBD or PSB.

---

**FABX0562E**      **VSAM *macro\_name* FAILED WITH  
RC=*rc* AND RSN=*rsn*. DD:*ddname***

### Explanation

The macro for the VSAM data set *ddname* DD failed. *rc* is the return code and *rsn* is the reason code from the macro.

### System action

IMS Library Integrity Utilities ends with a return code of 8.

## User response

See *z/OS DFSMS Macro Instructions for Data Sets* to determine the meaning of the return code and reason code.

Correct the error and rerun the job. If the error is not in the data set or your system, contact IBM Software Support.

---

**FABX0563E**      *keyword* **KEYWORD MUST BE SPECIFIED FOR THE SSID1 AND SSID2 STATEMENTS.**

## Explanation

The indicated keyword must be specified for the indicated statements.

## System action

IMS Library Integrity Utilities ends with a return code of 99.

## User response

Add the keyword to the indicated statements, and then rerun the job.

---

**FABX0565E**      *keyword1* **AND** *keyword2* **CANNOT BE SPECIFIED AT THE SAME TIME.**

## Explanation

An invalid parameter is specified for the keyword.

## System action

IMS Library Integrity Utilities ends with a return code of 8.

## User response

Correct the keyword parameter and rerun the job. For the supported combinations of keywords, see [“Control statements for the Catalog Manager utility”](#) on page 292.

---

**FABX0566I**      *DBD resource* **[ACTIVE|PENDING|\*\*\*\*\*] DBVER=number GENDATE=mm/dd/yyyy hh:mm:ss.th IS REFERRED.**

## Explanation

This message shows the progress that the DBD or PSB refers to an external DBD. If the status is neither ACTIVE nor PENDING, \*\*\*\*\* is shown.

## System action

IMS Library Integrity Utilities continues processing.

## User response

None. This message is informational.

---

**FABX0567W**      *DBD resource1* **IS NOT REFERRED BY [DBD|PSB] resource2.**

## Explanation

The DBD or PSB *resource2* being processed does not refer to its related DBD *resource1* because the Catalog Manager utility cannot identify the *resource1* instance from the IMS catalog database when one of the following conditions is met:

- *resource2* is not an Active or Pending instance.
- When *resource2* has a database versioning number and some of the DBD instances with the same database versioning number exist in the IMS catalog database, the timestamp of *resource2* is not the latest.
- When *resource2* is a LOGICAL DBD and some of the DBD instances of this LOGICAL DBD instance exist in the IMS catalog database, the timestamp of *resource2* is not the latest.

## System action

IMS Library Integrity Utilities continues processing without referencing *resource1*.

## User response:

None.

---

**FABX0568W**      **THE SPECIFIED [DBD|PSB] INSTANCE resource DOES NOT EXIST IN THE IMS CATALOG.**

## Explanation

The specified DBD or PSB instance was not found in the IMS catalog database.

## System action

The Catalog Manager utility continues processing.

## User response

Specify the correct resource name.

---

**FABX0569W**      *parameter* **\$FABXnnn ON statement STATEMENT PARAMETER IS NAMED AUTOMATICALLY**

## Explanation

\$FABX $nnn$  is automatically named by IMS Library Integrity Utilities because the parameter value on IMS catalog database is not valid.

## System action

IMS Library Integrity Utilities continues processing.

## User response

When you use the decoded DBDGEN source, you must set a valid value on \$FABX $nnn$ .

---

**FABX0570E**      **GETDSAB FAILED WITH RC= $rc$   
AND RSN= $rsn$ . DD= $ddname$**

## Explanation

The GETDSAB macro failed for the indicated DD. The return code is  $rc$ , and the reason code is  $rsn$ .

## System action

IMS Library Integrity Utilities ends with a return code of 8.

## User response

Find the cause of the error. The return code and reason code in the message are from the GETDSAB macro. Use the information about return codes and reason codes in the topic "GETDSAB description" in *z/OS MVS Programming: Authorized Assembler Services Reference, Volume 2 (EDTINFO-IXGWRITE)*. Correct the error, and rerun the job. If the error persists, contact IBM Software Support.

---

**FABX0571E**      **RDJFCB FAILED WITH RC= $rc$ .  
DD= $ddname$**

## Explanation

The RDJFCB macro failed for the indicated DD. The return code is  $rc$ .

## System action

IMS Library Integrity Utilities ends with a return code of 8.

## User response

Contact IBM Software Support.

---

**FABX0572I**      **FOUND [NO DBD RESOURCES |  
NO LOGICAL DBD AND GSAM DBD  
MEMBERS] IN THE DBD LIBRARY.**

## Explanation

This message is issued during export processing. Either the DBD library data set is empty, or no logical DBD and GSAM DBD members are found in the DBD library data set.

## System action

IMS Library Integrity Utilities continues processing.

## User response

None. This message is informational.

---

**FABX2001I**      **MDA REVERSAL ENDED  
NORMALLY.**

## Explanation

The MDA Reversal utility ended successfully.

## System action

The MDA Reversal utility ends with a return code of 0.

## User response

None. This message is informational.

---

**FABX2002W**      **MDA REVERSAL ENDED WITH  
WARNINGS.**

## Explanation

The MDA Reversal utility ended with warning conditions.

## System action

The MDA Reversal utility ends with a return code of 4.

## User response

Locate other FABX messages and identify the cause.

---

**FABX2003E**      **MDA REVERSAL ENDED WITH  
ERRORS.**

## Explanation

The MDA Reversal utility ended with errors.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

Locate other FABX messages and identify the cause.

---

**FABX2010I**      ***nnnn* DFSMDA MEMBERS  
SELECTED. *mmmm* DFSMDA  
MEMBERS PROCESSED.**

### Explanation

This message shows the number of DFSMDA members selected (*nnnn*) and the number of DFSMDA members processed (*mmmm*).

### System action

Processing continues.

### User response

None. This message is informational.

---

**FABX2020I**      **DFSMDA TYPE=FPDEDB IS  
DECODED AS TYPE=DATABASE  
BECAUSE OPTION FPDEDB\_LIB IS  
NOT SPECIFIED.**

### Explanation

Because OPTION FPDEDB\_LIB is not specified, the DFSMDA macro that originally had DFSMDA TYPE=FPDEDB is printed as DFSMDA TYPE=DATABASE. This message is issued even if DFSMDA TYPE=FPDEDB was not used to generate the original DFSMDA members.

### System action

The MDA Reversal utility continues processing.

### User response

Even if the TYPE parameter value is different, you can regenerate an identical DFSMDA member from the decoded DFSMDA macros. However, if you want the original TYPE parameter values printed in decoded DFSMDA macros, specify OPTION FPDEDB\_LIB for the MDA Reversal utility control statements and rerun the job.

---

**FABX2021I**      **DFSMDA TYPE=RECON WITH  
ALTERNATE DDNAME IS DECODED  
AS TYPE=DATABASE BECAUSE  
OPTION RECON\_ALT\_DD IS NOT  
SPECIFIED.**

### Explanation

Because OPTION RECON\_ALT\_DD is not specified, the DFSMDA macro that originally had DFSMDA TYPE=RECON, an alternate DD name, and WAIT=NO parameter is printed as DFSMDA TYPE=DATABASE. This message is issued even if DFSMDA TYPE=RECON

was not used to generate the original DFSMDA members.

### System action

The MDA Reversal utility continues processing.

### User response

Even if the TYPE parameter value is different, you can regenerate an identical DFSMDA member from the decoded DFSMDA macros. However, if you want the original TYPE parameter values printed in decoded DFSMDA macros, specify OPTION RECON\_ALT\_DD for the MDA Reversal utility control statements and rerun the job.

---

**FABX2103E**      **GETMAIN FAILED WITH RC=*rc*  
(SIZE=*size*).**

### Explanation

The GETMAIN macro failed.

### System action

The MDA Reversal utility ends with a return code of 8.

### 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.

---

**FABX2105E**      **OPEN FAILED FOR THE *ddname*  
DATA SET. RC=*rc*.**

### Explanation

The OPEN macro failed for the indicated DD.

### System action

The MDA Reversal utility ends with a return code of 8.

### User response

Check whether the correct data set is specified to the DD statement. Correct the error and rerun the job.

---

**FABX2106E**      **LOAD FAILED WITH SYSTEM  
COMPLETION CODE=*sc* AND  
RSN=*rsn*. MEMBER *member* IN  
*ddname* DD.**

## Explanation

The LOAD macro failed for a member in the indicated DD.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

Ensure that the correct member exists and rerun the job.

---

**FABX2108E**      **DESERV FAILED WITH RC=*rc* AND RSN=*rsn*. DD=*ddname***

## Explanation

The DESERV macro failed for the indicated DD.

## System action

The MDA Reversal 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 and reason code in the message are from the DESERV 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.

---

**FABX2109E**      ***ddname* DD IS MISSING.**

## Explanation

The indicated DD is not allocated.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

See "Control statements for the MDA Reversal utility" on page 272 and specify the indicated DD statement in the JCL.

---

**FABX2114E**      ***parameter* CANNOT BE SPECIFIED FOR THE *keyword* KEYWORD.**

## Explanation

An invalid parameter is specified for the indicated keyword.

## System action

The MDA Reversal 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 MDA Reversal utility" on page 272.

---

**FABX2116E**      **ERROR IN THE FABXMIN CONTROL STATEMENT.**

## Explanation

The MDA Reversal utility detected a control statement error in the FABXMIN control statement.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

Locate other WTO messages or refer to reports to identify the cause of the error. Then correct the error and rerun the job.

---

**FABX2117E**      ***statement* STATEMENT MUST NOT BE SPECIFIED MORE THAN ONCE.**

## Explanation

The indicated statement cannot be specified more than once.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

Correct the error and rerun the job.

---

**FABX2119E**      **PROC STATEMENT MUST BE SPECIFIED BEFORE THE *statement* STATEMENT.**

## Explanation

The PROC statement is not specified on the first line of the FABXMIN control statements.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

Specify the PROC statement on the first line in the FABXMIN data set, and then rerun the job.

---

**FABX2120E** *statement* **STATEMENT MUST BE SPECIFIED.**

## Explanation

The indicated statement is not specified in the FABXMIN control statements.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

See “Control statements for the MDA Reversal utility” on page 272 and specify the indicated statement. Then, rerun the job.

---

**FABX2121E** **SYNTAX ERROR IN THE CONTROL STATEMENT.**

## Explanation

The MDA Reversal utility detected a syntax error in the FABXMIN control statements.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

See “Control statements for the MDA Reversal utility” on page 272 and ensure that the control statement conforms to the syntax rules. Then, rerun the job.

---

**FABX2122E** **UNRECOGNIZED STATEMENT SPECIFIED. STATEMENT:**  
*statement*

## Explanation

The indicated statement is not a valid statement for the MDA Reversal utility.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

See “Control statements for the MDA Reversal utility” on page 272 and correct the indicated statement. Then, rerun the job.

---

**FABX2123E** *keyword* **KEYWORD CANNOT BE SPECIFIED FOR THE *statement* STATEMENT.**

## Explanation

The indicated keyword is not supported for the indicated statement.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

See “Control statements for the MDA Reversal utility” on page 272 and correct the control statement. Then, rerun the job.

---

**FABX2124E** **THE NUMBER OF *keyword* KEYWORDS EXCEEDED THE LIMIT. MAX IS *num*.**

## Explanation

The number of keywords that are specified in the FABXMIN data set exceeds the maximum number of keywords that can be specified.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

Specify fewer keywords, and then rerun the job.

---

**FABX2125E** *keyword* **KEYWORD MUST BE SPECIFIED FOR THE *statement* STATEMENT.**

## Explanation

The indicated keyword must be specified for the indicated statement.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

Add the keyword to the indicated statement, and then rerun the job.

---

**FABX2126E** **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 MDA Reversal utility ends with a return code of 8.

## User response

Specify fewer parameters, and then rerun the job.

---

<b>FABX2127E</b>	<b>THE <i>n</i>TH PARAMETER ON THE <i>keyword</i> KEYWORD HAS INCORRECT LENGTH.</b>
------------------	---

## Explanation

The length of the indicated parameter that is specified for the indicated keyword is invalid.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

See “Control statements for the MDA Reversal utility” on page 272 and correct the length of the indicated parameter. Then, rerun the job.

---

<b>FABX2133W</b>	<b>THE SPECIFIED DFSMDA MEMBER <i>member</i> DOES NOT EXIST IN THE LIBRARY.</b>
------------------	---

## Explanation

The indicated DFSMDA member was not found in the library of DFSMDA members.

## System action

The MDA Reversal utility skips the indicated member, sets the return code to 4, and continues processing.

## User response

Ensure that the indicated DFSMDA member is generated correctly and exists in the library. Then, rerun the job.

---

<b>FABX2150W</b>	<b>NO DFSMDA MEMBERS FOUND IN THE LIBRARY.</b>
------------------	--

## Explanation

The library is empty.

## System action

The MDA Reversal utility ends with a return code of 4.

## User response

Ensure that the MDA library is generated correctly and DFSMDA members exist in the library. Then, rerun the job.

---

<b>FABX2151W</b>	<b>DFSMDA MEMBER <i>mdaname</i> IS NOT A VALID DFSMDA MEMBER.</b>
------------------	---

## Explanation

The layout of the indicated DFSMDA member is invalid.

## System action

The MDA Reversal utility skips the indicated member, sets the return code to 4, and continues processing.

## User response

Ensure that the indicated DFSMDA member is generated correctly. Then, rerun the job.

---

<b>FABX2152W</b>	<b>LOAD FAILED WITH SYSTEM COMPLETION CODE=<i>sc</i> AND RSN=<i>rsn</i>. MEMBER <i>member</i> IN <i>ddname</i> DD.</b>
------------------	--

## Explanation

The MDA Reversal utility failed to load the indicated member.

## System action

The MDA Reversal utility skips the indicated member, sets the return code to 4, and continues processing.

## User response

Ensure that the indicated DFSMDA member is generated correctly. Then, rerun the job.

---

<b>FABX2153W</b>	<b>BLDL FAILED WITH RC=<i>rc</i>. MEMBER <i>member</i> IN <i>ddname</i> DD.</b>
------------------	---

## Explanation

The BLDL macro failed for a member in the indicated DD.

## System action

The MDA Reversal utility skips the indicated member, sets the return code to 4, and continues processing.

## User response

Ensure that the indicated DFSMDA member is generated correctly. Then, rerun the job.

---

**FABX2160E**      **WRITE FAILED. MEMBER *member* IN *ddname* DD.**

## Explanation

The WRITE macro failed for a member in the indicated DD.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

Determine the cause of the WRITE macro failure, correct the error, and rerun the job.

---

**FABX2161E**      **STOW FAILED WITH RETURN CODE=*rtn* AND RSN=*rsn*. MEMBER *member* IN *ddname* DD.**

## Explanation

The STOW macro failed for a member in the indicated DD.

## System action

The MDA Reversal utility ends with a return code of 8.

## User response

Determine the cause of the STOW macro failure. For the return code and the reason code, see the topic "STOW completion codes" in *z/OS DFSMS Macro Instructions for Data Sets*. Correct the problem, and rerun the job.

---

**FABX2170W**      **DFSMDA TYPE=FPDEDB IS DECODED AS TYPE=DATABASE BECAUSE ERRORS OCCURRED WHILE OBTAINING THE NAMES OF DEDB AREAS FROM THE ACBLIB.**

## Explanation

Errors occurred while obtaining the names of DEDB areas from the ACB library.

## System action

The MDA Reversal utility sets the return code to 4 and continues processing.

## User response

Although FPDEDB=ACBLIB option is specified on the OPTION statement, all or some DFSMDA members that were generated with DFSMDA TYPE=FPDEDB might be decoded as DFSMDA TYPE=DATABASE. If you want those members decoded as DFSMDA TYPE=FPDEDB, locate message FABX2171W and follow the user response for that message.

---

**FABX2171W**      **REASON: *message\_id* *message\_text***

## Explanation

This message accompanies message FABX2170W and indicates the reason of the errors that occurred while obtaining DEDB area names from the ACB library. The message ID (*message\_id*) indicates the error message issued by the ACB Reversal utility, which was called by the MDA Reversal utility.

## System action

The MDA Reversal utility sets the return code to 4 and continues processing.

## User response

See the explanation for the indicated ACB Reversal utility message and follow the user response for that message.

---

**FABX3901E**      **OPEN FAILED FOR DATA SET DD: *ddname* 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: *ddname* 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 maintenance level of IMS Library Integrity Utilities is too low. Apply the latest PTF.

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 *IMS Tools Base Configuration Guide*.

## 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 Documentation.

For more information about increasing the JCL REGION parameter, see the topic "REGION parameter" in the *z/OS MVS JCL Reference*.

---

**FABX3904E**      **ddname 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 IMS Administration Foundation or Distributed Access Infrastructure might be incomplete. See the *Overview and Customization* guide of the solution packs and complete the installation and customization steps.

---

**FABX3906E**      **THE DATABASE ORGANIZATION**  
                  **(organization) OF DATABASE**  
                  **dbdname 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.

## 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 520](#).

- 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.

```
//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.

<b>FABU1001I</b>	<b>DIAG ENDED NORMALLY</b>	<b>Explanation</b>
		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.

### 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.

---

**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 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 8 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 16. References

The following reference topics provide technical reference information for using IMS Library Integrity Utilities.

### Topics:

- [“Device and feature code tables” on page 527](#)
- [“Sample library members” on page 529](#)
- [“How to read syntax diagrams” on page 530](#)

## 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 35. Device code table*

Device code	Device	Device code	Device
00	3270,1	21	DPM-B01
01	3270P,1	22	DPM-B02
02	3270,2	23	DPM-B03
03	3270P,2	24	DPM-B04
04	274X	25	DPM-B05
05	FIDS	26	DPM-B06
06	FIDS3	27	DPM-B07
07	FIDS4	28	DPM-B08
08	FIN	29	DPM-B09
09	FIJP	2A	DPM-B10
0A	FIPB	2B	DPM-B11
0B	FIFP	2C	DPM-B12
0C	SCS1	2D	DPM-B13
0D	SCS2	2E	DPM-B14
0E	FIDS7	2F	DPM-B15
11	DPM-A01	41	3270-A01
12	DPM-A02	42	3270-A02
13	DPM-A03	43	3270-A03
14	DPM-A04	44	3270-A04
15	DPM-A05	45	3270-A05
16	DPM-A06	46	3270-A06
17	DPM-A07	47	3270-A07

Table 35. Device code table (continued)

Device code	Device	Device code	Device
18	DPM-A08	48	3270-A08
19	DPM-A09	49	3270-A09
1A	DPM-A10	4A	3270-A10
1B	DPM-A11	4B	3270-A11
1C	DPM-A12	4C	3270-A12
1D	DPM-A13	4D	3270-A13
1E	DPM-A14	4E	3270-A14
1F	DPM-A15	4F	3270-A15

### Feature code table

Table 36. Feature code table

Feature code	Feature
01	FEAT=1
02	FEAT=2
03	FEAT=3
04	FEAT=4
05	FEAT=5
06	FEAT=6
07	FEAT=7
08	FEAT=8
09	FEAT=9
0A	FEAT=10
40	FEAT=120
4A	FEAT=(NOCD,DEKYBD,PEN)
4B	FEAT=(CARD,DEKYBD,PEN)
50	FEAT=126
60	FEAT=132
7F	FEAT=IGNORE
C1	FEAT=(CARD,NOPFK,NOPEN)
C2	FEAT=(NOCD,NOPFK,PEN)
C3	FEAT=(CARD,NOPFK,PEN)
C4	FEAT=(NOCD,PFK,NOPEN)
C5	FEAT=(NOCD,PFK,NOPEN)
C6	FEAT=(NOCD,PFK,PEN)

Table 36. Feature code table (continued)

Feature code	Feature
C7	FEAT=(CARD,PFK,PEN)
C8	FEAT=(NOCD,DEKYBD,NOPEN)
C9	FEAT=(CARD,DEKYBD,NOPEN)

## 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 37. Sample JCL in the SHPSJCL0 library

Utility	Member	Description
All utilities	FABLLINK	Link-edits the IMS Library Integrity Utilities load modules.
Integrity Checker	FABLIVP3	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.
	FABLALSC	Creates alias name DSPCRTR0 for the FABLRTR0 module.
	FABLALSD	Deletes the alias name DSPCRTR0 from the FABLRTR0 module.
	FABLINIT	Initializes the Integrity Checker utility by creating a LICON data set, initializing the LICON data set, and creating a global option module.
FABLUMD1	Runs SMP/E RECEIVE/APPLY of USERMOD to install the FABLRTR0 module into the IMS SDFSRESL library.	
Consistency Checker	FABLIVP2	Runs the Consistency Checker utility.
Multiple Resource Checker	FABWIVP	Runs the Multiple Resource Checker utility.
DBD/PSB/ACB Compare, Mapper, Reversal	FABLIVP1	Runs the DBD/PSB/ACB Compare, Mapper, and Reversal utilities.
MDA Reversal	FABXMIVP	Runs the MDA Reversal utility.
Catalog Manager	FABXCIVP	Runs the Catalog Manager utility.
Advanced ACBGEN and ACBLIB Analyzer	FABQIVP	Runs the Advanced ACBGEN utility and the ACBLIB Analyzer utility.
	FABLQUMD1	Deletes alias DFSUACB0 from the SHPSLMD0 library of IMS Library Integrity Utilities and the LMOD entry of IMS Library Integrity Utilities SMP/E CSI.
	FABLQUMD2	Runs SMP/E LIST for the IMS DFSRRA80 source entry.
FABLQUMD3	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.	

Table 37. Sample JCL in the SHPSJCL0 library (continued)

Utility	Member	Description
MFS Reversal and Compare	FABVIVP	Runs the MFS Reversal utility and the MFS Compare utility.

The following tables summarize the members in the SHPSSAMP library.

Table 38. Sample JCL in the SHPSSAMP library

Utility	Member	Description
Integrity Checker	FABLCNV2	Migrates the LICON data set that is used in IMS Library Integrity Utilities 1.1.
DBD/PSB/ACB Reversal	FABNDFL1	Runs the DBD/PSB/ACB Reversal Site Default Generation utility to generate a SYSIN site default table.
	FABNDFL2	Runs the DBD/PSB/ACB Reversal Site Default Generation utility to report on the SYSIN site default table.
IMS Administration Tool	FAB\$TL01	Used for the Run IMS Utilities feature (JCL generation) of IMS Administration Tool to generate JCL for the Consistency Checker utility.
	FAB\$TL02	Used for the Run IMS Utilities feature (JCL generation) of IMS Administration Tool to generate JCL for the DBD/PSB Mapper utility.
	FAB\$TL03	Used for the Run IMS Utilities feature (JCL generation) of IMS Administration Tool to generate JCL for the ACB Mapper utility.

Table 39. Sample procedure in the SHPSSAMP library

Utility	Member	Description
Integrity Checker	FABLPGEN	This member contains the procedure for creating a global option module for the Integrity Checker utility.

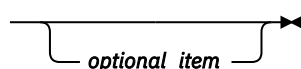
## How to read syntax diagrams

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

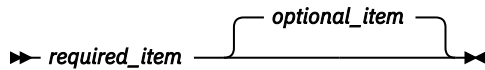
- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following conventions are used:
  - The >>--- symbol indicates the beginning of a syntax diagram.
  - The ---> symbol indicates that the syntax diagram is continued on the next line.
  - The >--- symbol 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.

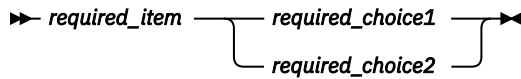
▶▶ *required\_item*  ◀◀

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

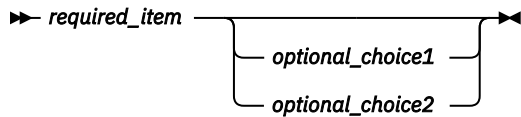


- If you can choose from two or more items, they appear vertically, in a stack.

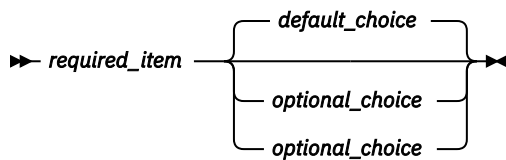
If you *must* choose one of the items, one item of the stack appears on the main path.



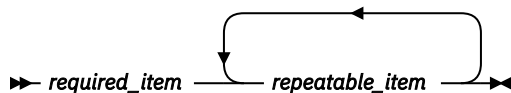
If choosing one of the items is optional, the entire stack appears below the main path.



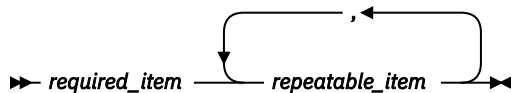
If one of the items is the default, it appears above the main path, and the remaining choices are shown below.



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

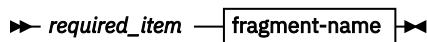


If the repeat arrow contains a comma, you must separate repeated items with a comma.

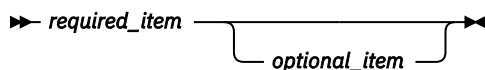


A repeat arrow above a stack indicates that you can repeat the items in the stack.

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



**fragment-name**



- A b symbol indicates one blank position.
- Keywords, and their minimum abbreviations if applicable, appear in uppercase. They must be spelled exactly as shown. Variables appear in all lowercase italic letters (for example, *column-name*). They represent user-supplied names or values.

- Separate keywords and parameters by at least one space if no intervening punctuation is shown in the diagram.
- Enter punctuation marks, parentheses, arithmetic operators, and other symbols exactly as shown in the diagram.
- Footnotes are shown by a number in parentheses; for example, (1).

## Notices

---

This information was developed for products and services offered in the U.S.A.

This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan Ltd.  
19-21, Nihonbashi-Hakozakicho, Chuo-ku  
Tokyo 103-8510, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites 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

Armonk, NY 10504-1785  
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

#### 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](http://www.ibm.com)<sup>®</sup> 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.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at <http://www.ibm.com/privacy> and the section titled "Cookies, Web Beacons, and Other Technologies" in IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details>. Also, see the "IBM Software Products and Software-as-a-Service Privacy Statement" at <http://www.ibm.com/software/info/product-privacy>.



# Index

## A

abend codes [399](#)  
ACB (DBD) map [210](#)  
ACB (DBD) report (DBD/PSB/ACB Mapper) [216](#)  
ACB (PSB) map [210](#)  
ACB (PSB) report (DBD/PSB/ACB Mapper) [216](#)  
ACB (PSB) Summary report (DBD/PSB/ACB Mapper) [214](#)  
ACB Compare report (DBD/PSB/ACB Compare) [187](#)  
ACB Generation and Catalog Populate utility [335](#)  
ACB Library Members Deleted Due to User Request report (Advanced ACBGEN) [345](#)  
ACB/PSB/DBD Library Information report (Advanced ACBGEN) [344](#)  
ACBCATWK DD (Advanced ACBGEN) [336](#)  
ACBGEN command (Advanced ACBGEN) [339](#)  
ACBLIB Analyzer  
  input [357](#)  
  JCL requirements [355](#)  
  output [358](#)  
  overview [355](#)  
  run [355](#)  
ACBLIB DD  
  Catalog Manager [288](#)  
  Consistency Checker [116](#)  
  DBD/PSB/ACB Compare [170](#)  
  DBD/PSB/ACB Mapper [203](#)  
  DBD/PSB/ACB Reversal [229](#)  
  LICON utility [86](#)  
  MDA Reversal [271](#)  
ACBLIB2 DD (DBD/PSB/ACB Compare) [170](#)  
ACBLIBxx DD (Multiple Resource Checker) [146](#)  
ACBSYSIN control statement  
  syntax rules for (Advanced ACBGEN) [339](#)  
ACBSYSIN DD  
  ACBLIB Analyzer [355](#)  
  Advanced ACBGEN [336](#)  
accessibility  
  keyboard shortcuts [23](#)  
  overview [23](#)  
ACTVPID keyword (MFS Compare) [385](#)  
ACTVPID name (MFS Reversal) [375](#)  
Advanced ACBGEN  
  input [338](#)  
  JCL requirements [336](#)  
  merging modules [334](#)  
  output [342](#)  
  overview [333](#)  
  run [334](#)  
ALPHA statement (MFS Reversal) [379](#)  
ATTR keyword (MFS Compare) [383](#), [385](#)

## B

BUILD control statement (Advanced ACBGEN) [339](#)  
BUILD PSB=ALL (Advanced ACBGEN) [344](#)

## C

CARD keyword (MFS Compare) [383](#)  
CARD name (MFS Reversal) [375](#)  
Catalog Manager  
  examples  
    compare [294](#)  
    convert [298](#)  
    generate maps [301](#)  
    validate [293](#)  
  examplescompare [304](#)  
  examplesconvert [308](#)  
  examplesgenerate maps [308–310](#)  
  examplesvalidate [304](#)  
  input  
    compare [294](#)  
    convert [298](#)  
    generate maps [301](#)  
    validate [293](#)  
  JCL requirements [288](#)  
  output  
    compare [317](#)  
    convert [321](#)  
    DBD map [326](#)  
    DBD report [329](#)  
    generate maps [324](#)  
    PSB map [326](#)  
    PSB report [329](#)  
    PSB Summary report [331](#)  
    validate [311](#)  
  overview [281](#)  
  run [286–288](#)  
CELLSIZE keyword (MFS Compare) [386](#)  
CHANGE.DB command (LICON utility) [95](#)  
CHECKBAT keyword (LICON utility) [91](#)  
CHECKIC keyword (LICON utility) [92](#)  
CHECKLD keyword (LICON utility) [91](#)  
CHECKON keyword (LICON utility) [91](#)  
CHKONLY keyword (Multiple Resource Checker) [148](#)  
Chronological History of ACBGENs report (ACBLIB Analyzer) [364](#)  
COMPCTL DD (Advanced ACBGEN) [336](#)  
COMPR keyword (MFS Compare) [384](#)  
COND keyword (MFS Compare) [382](#), [384](#)  
configuring  
  Consistency Checker [26](#)  
  Multiple Resource Checker [26](#)  
Consistency Checker  
  control statements [118](#)  
  data flow [113](#)  
  DBD Check report [122](#)  
  examples [120](#)  
  FABLECHK [113](#)  
  input [118](#)  
  JCL requirements [116](#)  
  output  
    SYSOUT data set [121](#)

- Consistency Checker (*continued*)
  - output (*continued*)
    - SYSPRINT data set [122](#)
  - overview [113](#)
  - PSB Check report [128](#)
  - restrictions [115](#)
  - run [115](#)
- contents of ACBSYSIN control statement data set
  - ACBLIB Analyzer [359](#)
  - Advanced ACBGEN [343](#)
- contents of EXEC statement parameter field
  - Advanced ACBGEN [343](#)
- contents of SYSIN control statement data set
  - Advanced ACBGEN [343](#)
- control statement source (DBD/PSB/ACB Reversal) [243](#)
- control statements
  - ACBLIB Analyzer [357](#)
  - Advanced ACBGEN [338](#)
  - Catalog Manager [292](#)
  - Consistency Checker [118](#)
  - DBD/PSB/ACB Compare [171](#)
  - DBD/PSB/ACB Mapper [204](#)
  - DBD/PSB/ACB Reversal [231](#)
  - FABLPGEN [80](#)
  - LICON utility [85](#)
  - MDA Reversal [272](#)
  - MFS Compare [389](#)
  - MFS Reversal [370](#)
  - Multiple Resource Checker [148](#)
- cookie policy [533](#)
- COPYFMT data set (MFS Reversal) [377](#)
- COPYFMT DD (MFS Reversal) [368](#)
- COPYPRT data set (MFS Reversal) [377](#)
- COPYPRT DD (MFS Reversal) [368](#)
- creating a new DBD (DBD/PSB/ACB Mapper) [208](#)
- creating a new PSB (DBD/PSB/ACB Mapper) [208](#)
- creating DBD, PSB, and ACB Compare reports (DBD/PSB/ACB Compare) [184](#)
- cross-reference report (MFS Reversal) [370](#), [373](#)
- CURSOR keyword (MFS Compare) [385](#)
- CURSOR name (MFS Reversal) [375](#)

## D

- data flow
  - Catalog Manager [281](#)
  - Consistency Checker [113](#)
  - DBD/PSB/ACB Compare [167](#)
  - DBD/PSB/ACB Mapper [201](#)
  - DBD/PSB/ACB Reversal [225](#)
  - MDA Reversal [269](#)
  - Multiple Resource Checker [135](#)
- DATA keyword (MFS Compare) [386](#)
- data set group information of DBD (DBD/PSB/ACB Mapper) [220](#)
- database access recording option [33](#)
- database information of DBD (Catalog Manager) [329](#)
- database information of DBD (DBD/PSB/ACB Mapper) [216](#)
- DBD
  - data set group information (DBD/PSB/ACB Mapper) [220](#)
  - database information (Catalog Manager) [329](#)
  - database information (DBD/PSB/ACB Mapper) [216](#)
  - field information (DBD/PSB/ACB Mapper) [222](#)

- DBD (*continued*)
  - library information (DBD/PSB/ACB Reversal) [242](#)
  - map (Catalog Manager) [326](#)
  - map (DBD/PSB/ACB Mapper) [210](#)
  - report (Catalog Manager) [329](#)
  - report (DBD/PSB/ACB Mapper) [216](#)
  - segment information (DBD/PSB/ACB Mapper) [220](#)
- DBD Check report (Consistency Checker) [122](#)
- DBD Compare report (DBD/PSB/ACB Compare) [187](#)
- DBD keyword (Multiple Resource Checker) [148](#)
- DBD map (Catalog Manager) [326](#)
- DBD map (DBD/PSB/ACB Mapper) [210](#)
- DBD report (Catalog Manager) [329](#)
- DBD report (DBD/PSB/ACB Mapper) [216](#)
- DBD XREF by Access report
  - DBD XREF by Access - Access Order (DBD/PSB/ACB Reversal) [247](#)
  - DBD XREF by Access - DBD Name Order (DBD/PSB/ACB Reversal) [247](#)
- DBD, PSB, and ACB map functions [206](#)
- DBD/PSB Names Specified via SYSIN report (Advanced ACBGEN) [344](#)
- DBD/PSB/ACB Compare
  - examples [184](#), [187](#)
  - FABLCOMP [167](#)
  - input
    - ACB control statement [172](#)
    - DBD control statement [172](#)
    - NOCOMP control statement [175](#)
    - PSB control statement [172](#)
    - REPORT control statement [174](#)
    - SYSIN data set [171](#)
- JCL requirements [170](#)
- job steps [167](#)
- output
  - SYSOUT data set [187](#)
  - SYSPRINT data set [187](#)
- overview [167](#)
- run [169](#)
- DBD/PSB/ACB Mapper
  - examples [206](#)
  - FABMMAIN [201](#)
  - input
    - SYSIN data set [204](#)
  - JCL requirements [203](#)
  - job steps [201](#)
  - output
    - ACB (DBD) map [210](#)
    - ACB (DBD) report [216](#)
    - ACB (PSB) map [210](#)
    - ACB (PSB) report [216](#)
    - ACB (PSB) Summary report [214](#)
    - DBD map [210](#)
    - DBD report [216](#)
    - PSB map [210](#)
    - PSB report [216](#)
    - PSB Summary report [214](#)
    - SYSPRINT data set [210](#), [216](#)
  - overview [201](#)
  - run [203](#)
- DBD/PSB/ACB Reversal
  - examples [241](#)
  - FABNRVRS [225](#)
  - input

DBD/PSB/ACB Reversal (*continued*)

input (*continued*)

SYSIN data set [231](#)

JCL requirements [229](#)

output

DBDSRC data set [247](#)

MAPOUT data set [261](#)

OPTPRT data set [261](#)

PSBSRC data set [247](#)

SYSOUT data set [243](#)

SYSPRINT data set [247](#)

SYSPUNCH data set [244](#)

overview [225](#)

run [228](#)

Site Default Generation utility

overview [262](#)

DBDLIB DD

Catalog Manager [288](#)

Consistency Checker [116](#)

DBD/PSB/ACB Compare

[170](#)

DBD/PSB/ACB Mapper [203](#)

DBD/PSB/ACB Reversal [229](#)

LICON utility [86](#)

DBDLIB2 DD (DBD/PSB/ACB Compare) [170](#)

DBDLIBxx DD (Multiple Resource Checker) [146](#)

DBDSRC data set (DBD/PSB/ACB Reversal) [247](#)

DBDSRC DD (Catalog Manager) [288](#)

DBDSRC DD (DBD/PSB/ACB Reversal) [229](#)

DBRC procedures [53](#)

DBTSNAP DD (ACBLIB Analyzer) [355](#)

DD statements

ACBCATWK DD (Advanced ACBGEN) [336](#)

ACBLIB DD

Catalog Manager [288](#)

Consistency Checker [116](#)

DBD/PSB/ACB Compare

[170](#)

DBD/PSB/ACB Mapper [203](#)

DBD/PSB/ACB Reversal [229](#)

LICON utility [86](#)

MDA Reversal [271](#)

ACBLIB2 DD (DBD/PSB/ACB Compare) [170](#)

ACBLIBxx DD (Multiple Resource Checker) [146](#)

ACBSYSIN DD

ACBLIB Analyzer [355](#)

Advanced ACBGEN [336](#)

COMPCTL DD (Advanced ACBGEN) [336](#)

COPYFMT DD (MFS Reversal) [368](#)

COPYPRT DD (MFS Reversal) [368](#)

DBDLIB DD

Catalog Manager [288](#)

Consistency Checker [116](#)

DBD/PSB/ACB Compare

[170](#)

DBD/PSB/ACB Mapper [203](#)

DBD/PSB/ACB Reversal [229](#)

LICON utility [86](#)

DBDLIB2 DD (DBD/PSB/ACB Compare) [170](#)

DBDLIBxx DD (Multiple Resource Checker) [146](#)

DBDSRC DD (Catalog Manager) [288](#)

DBDSRC DD (DBD/PSB/ACB Reversal) [229](#)

DBTSNAP DD (ACBLIB Analyzer) [355](#)

DFS3PPRM DD (Advanced ACBGEN) [336](#)

DD statements (*continued*)

DFSHDBSC DD (Catalog Manager) [288](#)

DFSMDDA DD

MDA Reversal [271](#)

DFSMDDA DD (Consistency Checker) [116](#)

DFSPRINT DD (Advanced ACBGEN) [336](#)

DFSRESLB DD

Advanced ACBGEN [336](#)

Catalog Manager [288](#)

Consistency Checker [116](#)

LICON utility [86](#)

DFSVSAMP DD (Catalog Manager) [288](#)

FABLICON DD (LICON utility) [86](#)

FABLIN DD (LICON utility) [86](#)

FABLPRNT DD

Integrity Checker [53](#)

LICON utility [86](#)

FABLSNAP DD (Integrity Checker) [53](#)

FABQRPT DD (ACBLIB Analyzer) [355](#)

FABWCTL DD (Multiple Resource Checker) [146](#)

FABWOUT DD (Multiple Resource Checker) [146](#)

FABWRRPT DD (Multiple Resource Checker) [146](#)

FABWSUMM DD (Multiple Resource Checker) [146](#)

FABXCIN DD (Catalog Manager) [288](#)

FABXCRP0 DD (Catalog Manager) [288](#)

FABXCRP1 DD (Catalog Manager) [288](#)

FABXCRP2 DD (Catalog Manager) [288](#)

FABXCSRC DD (Catalog Manager) [288](#)

FABXMIN DD

MDA Reversal [271](#)

FABXMOUT DD

MDA Reversal [271](#)

FABXMRPT DD

MDA Reversal [271](#)

FABXMSRC DD

MDA Reversal [271](#)

FABXPPRM DD (Catalog Manager) [288](#)

FORMAT DD

MFS Compare [387](#)

MFS Reversal [368](#)

FORMAT2 DD (MFS Compare) [387](#)

IMS DD

Advanced ACBGEN [336](#)

Catalog Manager [288](#)

IMSACB DD

ACBLIB Analyzer [355](#)

Advanced ACBGEN [336](#)

IMSACB01 DD (Advanced ACBGEN) [336](#)

IMSVnn DD (Multiple Resource Checker) [146](#)

JOBLIB DD

Catalog Manager [288](#)

DBD/PSB/ACB Compare [170](#)

DBD/PSB/ACB Mapper [203](#)

DBD/PSB/ACB Reversal [229](#)

LICON utility [86](#)

MDA Reversal [271](#)

Multiple Resource Checker [146](#)

MAPOUT DD (DBD/PSB/ACB Reversal) [229](#)

MDASRC DD

MDA Reversal [271](#)

MFSSRCE DD

MFS Compare [387](#)

MFS Reversal [368](#)

MFSSRCE2 DD (MFS Compare) [387](#)

DD statements (*continued*)

MODBLKS DD (Consistency Checker) [116](#)  
NSYSRDDS DD (Consistency Checker) [116](#)  
OPTPRT DD (DBD/PSB/ACB Reversal) [229](#)  
PROCLIB DD  
    Advanced ACBGEN [336](#)  
    Catalog Manager [288](#)  
PSBLIB DD  
    Catalog Manager [288](#)  
    Consistency Checker [116](#)  
    DBD/PSB/ACB Compare [170](#)  
    DBD/PSB/ACB Mapper [203](#)  
    DBD/PSB/ACB Reversal [229](#)  
PSBLIB2 DD (DBD/PSB/ACB Compare) [170](#)  
PSBLIBxx DD (Multiple Resource Checker) [146](#)  
PSBSRC DD (DBD/PSB/ACB Reversal) [229](#)  
RECONx DD  
    Catalog Manager [288](#)  
    Consistency Checker [116](#)  
    LICON utility [86](#)  
RECONxxn DD (Multiple Resource Checker) [146](#)  
SORT DD (ACBLIB Analyzer) [355](#)  
STEPLIB DD  
    ACBLIB Analyzer [355](#)  
    Advanced ACBGEN [336](#)  
    Catalog Manager [288](#)  
    Consistency Checker [116](#)  
    DBD/PSB/ACB Compare [170](#)  
    DBD/PSB/ACB Mapper [203](#)  
    DBD/PSB/ACB Reversal [229](#)  
    Integrity Checker [53](#)  
    LICON utility [86](#)  
    MDA Reversal [271](#)  
    MFS Compare [387](#)  
    MFS Reversal [368](#)  
    Multiple Resource Checker [146](#)  
SYSABEND DD (Catalog Manager) [288](#)  
SYSIN DD  
    Advanced ACBGEN [336](#)  
    Consistency Checker [116](#)  
    DBD/PSB/ACB Compare [170](#)  
    DBD/PSB/ACB Mapper [203](#)  
    DBD/PSB/ACB Reversal [229](#)  
    FABLPGEN program [80](#)  
    MFS Compare [387](#)  
    MFS Reversal [368](#)  
SYSLIB DD (FABLPGEN program) [80](#)  
SYSMOD DD (FABLPGEN program) [80](#)  
SYSMDUMP DD (Catalog Manager) [288](#)  
SYSOUT DD  
    Consistency Checker [116](#)  
    DBD/PSB/ACB Compare [170](#)  
    DBD/PSB/ACB Mapper [203](#)  
    DBD/PSB/ACB Reversal [229](#)  
    MFS Reversal [368](#)  
SYSPRINT DD  
    ACBLIB Analyzer [355](#)  
    Advanced ACBGEN [336](#)  
    Consistency Checker [116](#)  
    DBD/PSB/ACB Compare [170](#)  
    DBD/PSB/ACB Mapper [203](#)

DD statements (*continued*)

SYSPRINT DD (*continued*)  
    DBD/PSB/ACB Reversal [229](#)  
    MFS Compare [387](#)  
    MFS Reversal [368](#)  
    Multiple Resource Checker [146](#)  
SYSPUNCH DD (DBD/PSB/ACB Reversal) [229](#)  
SYSRDDS DD (Consistency Checker) [116](#)  
SYSUDUMP DD (Catalog Manager)  
    PSBSRC DD (Catalog Manager) [288](#)  
SYSUT3 DD (Advanced ACBGEN) [336](#)  
SYSUT4 DD (Advanced ACBGEN) [336](#)  
SYSYOUT DD  
    MFS Compare [387](#)  
DEFN option (MFS Reversal) [380](#)  
DELETE control statement (Advanced ACBGEN) [339](#)  
DELETE.DB command (LICON utility) [100](#)  
device characteristics table (MFS Reversal) [367](#), [370](#)  
device code table [527](#)  
DFLD label (MFS Reversal) [375](#), [380](#)  
DFLDNAME keyword (MFS Compare) [383](#)  
DFS Messages Summary report (Advanced ACBGEN) [349](#)  
DFS3PPRM DD (Advanced ACBGEN) [336](#)  
DFSHDBSC DD (Catalog Manager) [288](#)  
DFSMDA DD  
    MDA Reversal [271](#)  
DFSMDA DD (Consistency Checker) [116](#)  
DFSnnnn messages (Advanced ACBGEN) [336](#)  
DFSPPRM DD (Advanced ACBGEN) [336](#)  
DFSRESLB DD  
    Advanced ACBGEN [336](#)  
    Catalog Manager [288](#)  
    Consistency Checker [116](#)  
    LICON utility [86](#)  
DFSUACB0 [333](#)  
DFSUDT0 table (MFS Reversal) [367](#)  
DFSVSAMP DD (Catalog Manager) [288](#)  
diagnostic information  
    gathering [520](#)  
diagnostics aid [520](#)  
DIF/DOF control blocks (MFS Reversal) [367](#)  
DIFREP keyword (Multiple Resource Checker) [148](#)  
directory information (ACBLIB Analyzer) [359](#)  
Distribution of Member Sizes report (ACBLIB Analyzer) [363](#)  
Distribution of PSB Workarea Sizes report (ACBLIB Analyzer) [364](#)  
DIV statement (MFS Reversal) [380](#)  
DMB verification  
    restrictions [70](#)  
DMB verification option [71](#)  
documentation  
    accessing [21](#)  
    sending feedback [21](#)  
double-step verification [71](#), [80](#)  
DPAGE label (MFS Reversal) [375](#)  
DPMA devices (MFS Reversal) [380](#)  
DPMB devices (MFS Reversal) [380](#)  
DPN keyword (MFS Compare) [384](#)  
DSCA keyword (MFS Compare) [383](#)  
DSCA value (MFS Reversal) [379](#)  
DVCTBL control statement  
    MFS Compare [389](#)  
    MFS Reversal [370](#)

## E

### EATTR

MFS Compare [385](#)

MFS Reversal [379](#)

EGCS specification (MFS Reversal) [379](#)

EJECT option (MFS Reversal) [380](#)

ENDMSG (MFS Reversal) [380](#)

### examples

Catalog Manager

compare [294](#)

convert [298](#)

generate maps [301](#)

validate [293](#)

Catalog Managercompare [304](#)

Catalog Managerconvert [308](#)

Catalog Managergenerate maps [308–310](#)

Catalog Managervalidate [304](#)

Consistency Checker [120](#)

DBD/PSB/ACB Compare [184](#), [187](#)

DBD/PSB/ACB Mapper [206](#)

DBD/PSB/ACB Reversal [241](#)

MDA Reversal [272](#), [275](#)

Multiple Resource Checker [155](#)

EXCLUDE control statement

MFS Reversal [370](#)

EXEC statement

ACBLIB Analyzer [355](#)

Advanced ACBGEN [336](#)

Catalog Manager [288](#)

DBD/PSB/ACB Compare

[170](#)

DBD/PSB/ACB Mapper [203](#)

DBD/PSB/ACB Reversal [229](#)

LICON utility [86](#)

MDA Reversal [271](#)

MFS Compare [387](#)

MFS Reversal [368](#)

EXIT keyword (MFS Compare) [383](#)

EXPIRE.DB command (LICON utility) [101](#)

## F

FABLCOMP (Compare utility) [167](#)

FABLECHK (Consistency Checker) [113](#)

FABLICON DD (LICON utility) [86](#)

FABLIN DD (LICON utility) [86](#)

FABLPGEN control statement keywords [80](#)

FABLPGEN JCL requirements [80](#)

FABLPGEN procedure [48](#), [79](#)

FABLPGIN macro [80](#)

FABLPRNT DD

Integrity Checker [53](#)

LICON utility [86](#)

FABLSNAP DD (Integrity Checker) [53](#)

FABMMAIN [201](#)

FABNRVRS [225](#)

FABQRPT DD (ACBLIB Analyzer) [355](#)

FABVDVCT table (MFS Reversal) [367](#)

FABVRVRS (MFS Reversal) [367](#)

FABWCTL DD (Multiple Resource Checker) [146](#)

FABWOUT DD (Multiple Resource Checker) [146](#)

FABWRRPT DD (Multiple Resource Checker) [146](#)

FABWSUMM DD (Multiple Resource Checker) [146](#)

FABXCIN DD (Catalog Manager) [288](#)

FABXCRP0 DD (Catalog Manager) [288](#)

FABXCRP1 DD (Catalog Manager) [288](#)

FABXCRP2 DD (Catalog Manager) [288](#)

FABXCSRC DD (Catalog Manager) [288](#)

FABXMIN DD

MDA Reversal [271](#)

FABXMOUT DD

MDA Reversal [271](#)

FABXMRPT DD

MDA Reversal [271](#)

FABXMSRC DD

MDA Reversal [271](#)

FABXPPRM DD (Catalog Manager) [288](#)

FEAT keyword (MFS Compare) [383](#)

feature code table [527](#)

field information of DBD (DBD/PSB/ACB Mapper) [222](#)

FILL keyword (MFS Compare) [382](#), [383](#), [385](#)

Final PSB Build List report (Advanced ACBGEN) [345](#)

finance terminals (MFS Reversal) [380](#)

format control blocks

MFS Compare [387](#)

MFS Reversal [369](#)

FORMAT DD

MFS Compare [387](#)

MFS Reversal [368](#)

FORMAT2 DD (MFS Compare) [387](#)

FORMS keyword (MFS Compare) [384](#)

FTAB keyword (MFS Compare) [383](#)

## G

G'LITERAL' keyword (MFS Compare) [385](#)

GENMAX keyword (LICON utility) [93](#)

global option module

creating [79](#)

global option module generation macro [79](#)

global option modules

creating [48](#)

GRAPHIC keyword (MFS Compare) [383](#)

## H

hardware prerequisites [25](#)

HDRCTL (MFS Reversal) [380](#)

HDRCTL keyword (MFS Compare) [384](#)

HTAB keyword (MFS Compare) [384](#)

## I

IF label (MFS Reversal) [375](#)

IMS [10](#)

IMS DD

Advanced ACBGEN [336](#)

IMS DD (Catalog Manager) [288](#)

IMS messages [400](#)

IMSACB DD

ACBLIB Analyzer [355](#)

Advanced ACBGEN [336](#)

IMSACB01 DD (Advanced ACBGEN) [336](#)

IMSVnn DD (Multiple Resource Checker) [146](#)

INDD= operand (ACBLIB Analyzer) [358](#)

INIT.DB command (LICON utility) [89](#)

INIT.LICON command (LICON utility) [95](#)

input

- ACBLIB Analyzer [357](#)
- Advanced ACBGEN [338](#)
- Catalog Manager
  - compare [294](#)
  - convert [298](#)
  - generate maps [301](#)
  - validate [293](#)
- Consistency Checker [118](#)
- DBD/PSB/ACB Compare [171](#)
- DBD/PSB/ACB Mapper [204](#)
- DBD/PSB/ACB Reversal [231](#)
- LICON utility [88](#)
- MDA Reversal [272](#)
- MFS Compare [389](#)
- MFS Reversal [370](#)
- Multiple Resource Checker [148](#)

Input Specifications report (ACBLIB Analyzer) [359](#)

Input Specifications report (Advanced ACBGEN) [342](#)

Integrity Checker

- activating [48](#)
- addressing DMB mismatch [75](#)
- applying PTFs [69](#)
- BPE-based DBRC [57](#)
- configuration [38](#)
- considerations [46](#)
- database access recording option [44](#)
- deactivating [76](#)
- design configuration [36](#)
- global option modules
  - changing [67](#)
  - effective ranges [68](#)
- historical data in LICON data sets [44](#)
- LICON data sets [37](#), [49](#)
- LIU load module library [36](#)
- LIU load modules [53](#)
- maintaining [59](#)
- maintaining global option modules [67](#)
- maintaining LICON data sets [69](#)
- maintaining RDEs
  - database recovery [62](#)
  - database reorganization [60](#)
  - DBD change [63](#)
  - initial database load [59](#)
- overview [33](#)
- planning [36](#)
- preventing [70](#)
- program structures [33](#)
- RACF security [51](#)
- restart [58](#), [69](#)
- restrictions [70](#)
- runtime options [43](#)
- verifying activation [58](#)

## J

JCL requirements

- ACBLIB Analyzer [355](#)
- Advanced ACBGEN [336](#)
- Catalog Manager [288](#)
- Consistency Checker [116](#)
- DBD/PSB/ACB Compare [170](#)
- DBD/PSB/ACB Mapper [203](#)

JCL requirements (*continued*)

- DBD/PSB/ACB Reversal [229](#)
  - FABLIU00 [86](#)
  - FABLPGEN [80](#)
  - LICON utility [86](#)
  - MDA Reversal [271](#)
  - MFS Compare [387](#)
  - MFS Reversal [368](#)
  - Multiple Resource Checker [146](#)
- JOBLIB DD
- Catalog Manager [288](#)
  - DBD/PSB/ACB Compare [170](#)
  - DBD/PSB/ACB Mapper [203](#)
  - DBD/PSB/ACB Reversal [229](#)
  - LICON utility [86](#)
  - MDA Reversal [271](#)
  - Multiple Resource Checker [146](#)
- JUST keyword (MFS Compare) [383](#)

## K

keyboard shortcuts [23](#)

## L

LDEL keyword (MFS Compare) [383](#)

legal notices

- cookie policy [533](#)
- notices [533](#)
- programming interface information [533](#)
- trademarks [533](#)

LENGTH keyword (MFS Compare) [386](#)

libraries used (ACBLIB Analyzer) [359](#)

Library Contents report (ACBLIB Analyzer) [360](#)

library information of DBD (DBD/PSB/ACB Reversal) [242](#)

library information of PSB (DBD/PSB/ACB Reversal) [242](#)

Library Information report (ACBLIB Analyzer) [359](#)

LIBTYPE= operand (ACBLIB Analyzer) [358](#)

LICON data sets

- defining and initializing [49](#)
- estimating size [46](#)

LICON utility

- control statements [85](#)
- input [88](#)
- JCL requirements [86](#)
- output [106](#)
- runtime options [88](#)

LICON utility control statement keywords [85](#)

LIST.DB command (LICON utility) [102](#)

LIST.LICON command (LICON utility) [103](#)

LISTLIB command (ACBLIB Analyzer) [357](#)

LITERAL keyword (MFS Compare) [385](#)

LIU load module library [36](#)

Load Module Mgmt Stats report (Advanced ACBGEN) [352](#)

lower case characters (MFS Reversal) [379](#)

LTH keyword (MFS Compare) [383](#), [385](#)

LTH keyword (MFS Reversal) [379](#)

LUDEFN keyword (MFS Compare) [386](#)

LUSIZE keyword (MFS Compare) [386](#)

## M

map



map (*continued*)  
 ACB (DBD) map (DBD/PSB/ACB Mapper) [210](#)  
 ACB (PSB) map (DBD/PSB/ACB Mapper) [210](#)  
 DBD map (Catalog Manager) [326](#)  
 DBD map (DBD/PSB/ACB Mapper) [210](#)  
 PSB map (Catalog Manager) [326](#)  
 PSB map (DBD/PSB/ACB Mapper) [210](#)  
 MAPOUT data set (DBD/PSB/ACB Reversal) [261](#)  
 MAPOUT DD (DBD/PSB/ACB Reversal) [229](#)  
 mapper input (DBD/PSB/ACB Reversal) [243](#)  
 mapping a new DBD (DBD/PSB/ACB Mapper) [208](#)  
 mapping a new PSB (DBD/PSB/ACB Mapper) [208](#)  
 MDA Reversal  
 examples [272](#), [275](#)  
 input [272](#)  
 JCL requirements [271](#)  
 output  
 FABXMOUT data set [278](#)  
 FABXMRPT data set [279](#)  
 FABXMSRC data set [275](#)  
 MDASRC data set [278](#)  
 overview [269](#)  
 run [271](#)  
 MDASRC DD  
 MDA Reversal [271](#)  
 message descriptor codes [82](#)  
 message routing codes [82](#)  
 messages  
 Advanced ACBGEN [472](#)  
 Consistency Checker [401](#)  
 DBD/PSB Map Viewer [499](#)  
 DBD/PSB/ACB Compare [401](#)  
 DBD/PSB/ACB Mapper [446](#)  
 DBD/PSB/ACB Reversal [453](#)  
 FABL messages [401](#)  
 FABM messages [446](#)  
 FABN messages [453](#)  
 FABQ messages [472](#)  
 FABV messages [487](#)  
 FABW messages [495](#)  
 FABX messages [499](#)  
 Integrity Checker [401](#)  
 LICON utility [401](#)  
 methods for accessing [519](#)  
 MFS Compare [487](#)  
 MFS Reversal [487](#)  
 Multiple Resource Checker [495](#)  
 MFLD label (MFS Reversal) [375](#)  
 MFLD name (MFS Reversal) [375](#)  
 MFS Compare  
 JCL requirements [387](#)  
 overview [381](#)  
 report [390](#)  
 run [386](#)  
 MFS Reversal  
 copy function [367](#)  
 JCL requirements [368](#)  
 overview [367](#)  
 run [368](#)  
 MFSSRCE data set (MFS Reversal) [375](#)  
 MFSSRCE DD  
 MFS Compare [387](#)  
 MFS Reversal [368](#)  
 MFSSRCE2 DD (MFS Compare) [387](#)  
 MID/MOD control blocks (MFS Reversal) [367](#)  
 migration  
 Advanced ACBGEN [31](#)  
 Miscellaneous DFS Messages report (Advanced ACBGEN) [349](#)  
 MODBLKS DD (Consistency Checker) [116](#)  
 MODE keyword (MFS Compare) [383](#)  
 MONITOR= operand (Advanced ACBGEN) [339](#)  
 MULT keyword (MFS Compare) [385](#)  
 Multiple Resource Checker  
 checking consistencies [136](#)  
 control statements [148](#)  
 examples [155](#)  
 JCL requirements [146](#)  
 output  
 FABWOUT data set [159](#)  
 FABWRRPT data set [162](#)  
 FABWSUMM data set [159](#)  
 RECON Difference report [162](#)  
 Resource Check Summary report [159](#)  
 overview [135](#)

## N

naming conventions (MFS Reversal) [375](#)  
 new DBD (DBD/PSB/ACB Mapper) [208](#)  
 new PSB (DBD/PSB/ACB Mapper) [208](#)  
 NOCOMP keyword (Multiple Resource Checker) [148](#)  
 NOSPAN (MFS Reversal) [380](#)  
 notices [533](#)  
 NSYSRDDSD DD (Consistency Checker) [116](#)  
 NULL keyword (MFS Compare) [384](#)  
 NXT keyword (MFS Compare) [382](#)  
 NXT keyword (MFS Reversal) [379](#)

## O

OFTAB keyword (MFS Compare) [384](#), [385](#)  
 OPCTL keyword (MFS Compare) [385](#)  
 OPCTL TABLE label (MFS Reversal) [375](#)  
 OPT keyword (MFS Compare) [382](#)  
 OPTION control statement  
 MFS Reversal [370](#)  
 OPTIONS keyword (MFS Compare) [384](#)  
 OPTIONS=DPAGE (MFS Reversal) [380](#)  
 OPTIONS=MSG (MFS Reversal) [380](#)  
 OPTIONS=SIM (MFS Reversal) [380](#)  
 OPTPRT data set (DBD/PSB/ACB Reversal) [261](#)  
 OPTPRT DD (DBD/PSB/ACB Reversal) [229](#)  
 ORIGIN keyword (MFS Compare) [385](#)  
 OUTL specification (MFS Reversal) [379](#)  
 output  
 ACBLIB Analyzer utility [358](#)  
 Advanced ACBGEN [342](#)  
 Catalog Manager [311](#), [329](#)  
 Consistency Checker [121](#)  
 DBD/PSB/ACB Compare [187](#)  
 DBD/PSB/ACB Mapper [209](#),  
[216](#)  
 DBD/PSB/ACB Reversal [243](#)  
 LICON utility [106](#)  
 MDA Reversal [275](#)

output (*continued*)

MFS Compare [390](#)  
MFS Reversal [373](#)  
Multiple Resource Checker [159](#)

## P

PAGE keyword (MFS Compare) [382](#)  
PAGE keyword (MFS Reversal) [380](#)  
page size (MFS Reversal) [379](#)  
PAGESIZE= operand (Advanced ACBGEN) [339](#)  
PAGINGOP keyword (MFS Compare) [386](#)  
PASSWORD keyword (MFS Compare) [382](#), [385](#)  
PD keyword (MFS Compare) [385](#)  
PDB keyword (MFS Compare) [384](#)  
PEN keyword (MFS Compare) [383](#), [385](#)  
PEN name (MFS Reversal) [375](#)  
PFK keyword (MFS Compare) [383](#)  
PFK name (MFS Reversal) [375](#)  
PGE keyword (MFS Compare) [383](#)  
PID keyword (MFS Compare) [386](#)  
POS keyword (MFS Compare) [385](#)  
post-compression (Advanced ACBGEN) [344](#)  
PPAGE keyword (MFS Compare) [385](#)  
PPAGE name (MFS Reversal) [380](#)  
PPAGE statement (MFS Reversal) [380](#)  
pre-compression (Advanced ACBGEN) [344](#)  
PRESpace keyword (MFS Compare) [386](#)  
PRN keyword (MFS Compare) [384](#)  
problems  
    diagnostic information about [520](#)  
PROCLIB DD  
    Advanced ACBGEN [336](#)  
    Catalog Manager [288](#)  
program functions  
    overview [11](#)  
programming interface information [533](#)  
progress monitor requested (Advanced ACBGEN) [344](#)  
PROMPT keyword (MFS Compare) [382](#)  
PSB  
    library information (DBD/PSB/ACB Reversal) [242](#)  
    map (Catalog Manager) [326](#)  
    map (DBD/PSB/ACB Mapper) [210](#)  
    report (Catalog Manager) [329](#)  
    report (DBD/PSB/ACB Mapper) [216](#)  
    summary report (Catalog Manager) [331](#)  
    summary report (DBD/PSB/ACB Mapper) [214](#)  
PSB Check report (Consistency Checker) [128](#)  
PSB Compare report (DBD/PSB/ACB Compare) [187](#)  
PSB keyword (Multiple Resource Checker) [148](#)  
PSB map (Catalog Manager) [326](#)  
PSB map (DBD/PSB/ACB Mapper) [210](#)  
PSB report (Catalog Manager) [329](#)  
PSB report (DBD/PSB/ACB Mapper) [216](#)  
PSB Size Summary report (Advanced ACBGEN) [346](#)  
PSB Summary report (Catalog Manager) [331](#)  
PSB Summary report (DBD/PSB/ACB Mapper) [214](#)  
PSB XREF by Type report  
    PSB XREF by Type - PSB Name Order (DBD/PSB/ACB Reversal) [248](#)  
    PSB XREF by Type - Type Order (DBD/PSB/ACB Reversal) [248](#)  
PSB/DBD Change Summary report (Advanced ACBGEN) [347](#)

PSBLIB DD

Catalog Manager [288](#)  
Consistency Checker [116](#)  
DBD/PSB/ACB Compare  
    [170](#)  
DBD/PSB/ACB Mapper [203](#)  
DBD/PSB/ACB Reversal [229](#)

PSBLIB2 DD (DBD/PSB/ACB Compare) [170](#)  
PSBLIBxx DD (Multiple Resource Checker) [146](#)  
PSBSRC data set (DBD/PSB/ACB Reversal) [247](#)  
PSBSRC DD (Catalog Manager) [288](#)  
PSBSRC DD (DBD/PSB/ACB Reversal) [229](#)

## R

RCD keyword (MFS Compare) [385](#)  
RCD statement (MFS Reversal) [380](#)  
RCDCTL (MFS Reversal) [380](#)  
RCDCTL keyword (MFS Compare) [384](#)  
RDEs  
    creating [50](#)  
RDPN keyword (MFS Compare) [384](#)  
reader comment form [21](#)  
RECLD keyword (LICON utility) [93](#)  
RECON Difference report (Multiple Resource Checker) [162](#)  
RECONx DD  
    Catalog Manager [288](#)  
    Consistency Checker [116](#)  
    LICON utility [86](#)  
RECONxxn DD (Multiple Resource Checker) [146](#)  
RECOVER.DB command (LICON utility) [104](#)  
RECU keyword (LICON utility) [93](#)  
RECUPD keyword (LICON utility) [93](#)  
reports  
    ACB (DBD) report (DBD/PSB/ACB Mapper) [216](#)  
    ACB (PSB) report (DBD/PSB/ACB Mapper) [216](#)  
    ACB (PSB) Summary report (DBD/PSB/ACB Mapper) [214](#)  
    ACB Compare report (DBD/PSB/ACB Compare) [187](#)  
    ACB Library Members Deleted Due to User Request report (Advanced ACBGEN) [345](#)  
    ACB(DBD) to ACB(DBD) XREF report (DBD/PSB/ACB Reversal) [252](#)  
    ACB(DBD) XREF by Access report (DBD/PSB/ACB Reversal) [247](#)  
    ACB(DBD) XREF by DDname report (DBD/PSB/ACB Reversal) [249](#)  
    ACB(PSB) to ACB(DBD) XREF report (DBD/PSB/ACB Reversal) [254](#)  
    ACB(PSB) XREF by Type report (DBD/PSB/ACB Reversal) [248](#)  
    ACB/PSB/DBD Library Information report (Advanced ACBGEN) [344](#)  
    ACBLIB Analyzer [358](#)  
    Advanced ACBGEN [342](#)  
    Chronological History of ACBGENs report (ACBLIB Analyzer) [364](#)  
    DBD Check report (Consistency Checker) [122](#)  
    DBD Compare report (DBD/PSB/ACB Compare) [187](#)  
    DBD report (Catalog Manager) [329](#)  
    DBD report (DBD/PSB/ACB Mapper) [216](#)  
    DBD Segment Reference report (DBD/PSB/ACB Reversal) [256](#)  
    DBD to DBD XREF report (DBD/PSB/ACB Reversal) [252](#)

reports (*continued*)

DBD XREF by Access report (DBD/PSB/ACB Reversal) [247](#)  
DBD XREF by DDname report (DBD/PSB/ACB Reversal) [249](#)  
DBD/PSB Names Specified via SYSIN report (Advanced ACBGEN) [344](#)  
DFS Messages Summary report (Advanced ACBGEN) [349](#)  
Distribution of Member Sizes report (ACBLIB Analyzer) [363](#)  
Distribution of PSB Workarea Sizes report (ACBLIB Analyzer) [364](#)  
Final PSB Build List report (Advanced ACBGEN) [345](#)  
Input Specifications report (ACBLIB Analyzer) [359](#)  
Input Specifications report (Advanced ACBGEN) [342](#)  
Library Contents report (ACBLIB Analyzer) [360](#)  
Library Information report (ACBLIB Analyzer) [359](#)  
Load Module Management Status (Load Module Mgmt Stats) report (Advanced ACBGEN) [352](#)  
Miscellaneous DFS Messages report (Advanced ACBGEN) [349](#)  
PCB PROCOPT report (DBD/PSB/ACB Reversal) [251](#)  
PCB/ACB(PSB) PROCOPT report (DBD/PSB/ACB Reversal) [251](#)  
PSB Check report (Consistency Checker) [128](#)  
PSB Compare report (DBD/PSB/ACB Compare) [187](#)  
PSB PROCOPT reference reports for PSB and ACB(PSB) (DBD/PSB/ACB Reversal) [258](#)  
PSB report (Catalog Manager) [329](#)  
PSB report (DBD/PSB/ACB Mapper) [216](#)  
PSB Segment Reference report (DBD/PSB/ACB Reversal) [257](#)  
PSB Size Summary report (Advanced ACBGEN) [346](#)  
PSB Summary report (Catalog Manager) [331](#)  
PSB Summary report (DBD/PSB/ACB Mapper) [214](#)  
PSB to DBD XREF report (DBD/PSB/ACB Reversal) [254](#)  
PSB XREF by Type report (DBD/PSB/ACB Reversal) [248](#)  
PSB/DBD Change Summary report (Advanced ACBGEN) [347](#)  
RECON Difference report [162](#)  
Resource Check Summary report (Multiple Resource Checker) [159](#)  
Run Summary report (Advanced ACBGEN) [350](#)  
Unreferenced ACB(DBD) report (DBD/PSB/ACB Reversal) [259](#), [260](#)  
Warning Messages report (ACBLIB Analyzer) [365](#)  
REPORTS= operand (Advanced ACBGEN) [339](#)  
Resource Check Summary report (Multiple Resource Checker) [159](#)  
return codes [393](#)  
RPRN keyword (MFS Compare) [384](#)  
Run Summary report (Advanced ACBGEN) [350](#)  
running  
ACBLIB Analyzer [355](#)  
Advanced ACBGEN [334](#)  
Catalog Manager  
compare [286](#)  
convert [287](#)  
generate maps [288](#)  
validate [286](#)  
Consistency Checker [115](#)  
DBD/PSB/ACB Compare [169](#)  
DBD/PSB/ACB Mapper [203](#)

running (*continued*)

DBD/PSB/ACB Reversal [228](#)  
MDA Reversal [271](#)  
MFS Compare [386](#)  
MFS Reversal [368](#)  
Multiple Resource Checker [136](#)  
runtime options  
LICON utility [88](#)  
runtime parameters (Advanced ACBGEN) [343](#)

## S

sample generated source statements (MFS Reversal) [375](#)  
sample library [529](#)  
sample MFS Compare report [390](#)  
SCA keyword (MFS Compare) [383](#)  
screen readers and magnifiers [23](#)  
SCROLLI keyword (MFS Compare) [386](#)  
SCS1 devices (MFS Reversal) [380](#)  
segment information of DBD (DBD/PSB/ACB Mapper) [220](#)  
SELECT control statement  
MFS Compare [389](#)  
MFS Reversal [370](#)  
SELECT keyword (MFS Compare) [385](#)  
service information [21](#)  
SHPSJCL0 [529](#)  
SHPSSAMP [529](#)  
single-step verification [71](#), [80](#)  
SLDI (MFS Reversal) [380](#)  
SLDI keyword (MFS Compare) [384](#), [385](#)  
SLDP keyword (MFS Compare) [384](#), [385](#)  
SNAP= operand (ACBLIB Analyzer) [358](#)  
software prerequisites [25](#)  
SOR keyword (MFS Compare) [382](#)  
SORT DD (ACBLIB Analyzer) [355](#)  
source generated by utilities (MFS Reversal) [367](#)  
SPACE option (MFS Reversal) [380](#)  
STEPLIB DD  
ACBLIB Analyzer [355](#)  
Advanced ACBGEN [336](#)  
Catalog Manager [288](#)  
Consistency Checker [116](#)  
DBD/PSB/ACB Compare [170](#)  
DBD/PSB/ACB Mapper [203](#)  
DBD/PSB/ACB Reversal [229](#)  
Integrity Checker [53](#)  
LICON utility [86](#)  
MDA Reversal [271](#)  
MFS Compare [387](#)  
MFS Reversal [368](#)  
Multiple Resource Checker [146](#)  
SUB keyword (MFS Compare) [384](#)  
support  
required information [520](#)  
support information [21](#)  
suppressing output  
MFS Reversal [370](#)  
syntax diagrams  
how to read [530](#)  
syntax rules for ACBSYSIN control statements (Advanced ACBGEN) [339](#)  
SYSABEND DD (Catalog Manager) [288](#)  
SYSIN data set

SYSIN data set (*continued*)  
     DBD/PSB/ACB Compare [171](#)  
     DBD/PSB/ACB Mapper [204](#)  
     DBD/PSB/ACB Reversal [231](#)  
 SYSIN DD  
     Advanced ACBGEN [336](#)  
     Consistency Checker [116](#)  
     DBD/PSB/ACB Compare [170](#)  
     DBD/PSB/ACB Mapper [203](#)  
     DBD/PSB/ACB Reversal [229](#)  
     FABLPGEN program [80](#)  
     MFS Compare [387](#)  
     MFS Reversal [368](#)  
 SYSLIB DD (FABLPGEN program) [80](#)  
 SYSLMOD DD (FABLPGEN program) [80](#)  
 SYSMDUMP DD (Catalog Manager) [288](#)  
 SYSMSG keyword (MFS Compare) [383](#), [386](#)  
 SYSOUT data set  
     Consistency Checker [121](#)  
     DBD/PSB/ACB Compare [187](#)  
     DBD/PSB/ACB Mapper [209](#)  
     DBD/PSB/ACB Reversal [243](#)  
     MFS Compare [390](#)  
     MFS Reversal [373](#)  
 SYSOUT DD  
     Consistency Checker [116](#)  
     DBD/PSB/ACB Compare [170](#)  
     DBD/PSB/ACB Mapper [203](#)  
     DBD/PSB/ACB Reversal [229](#)  
     MFS Reversal [368](#)  
 SYSPRINT data set  
     Consistency Checker [122](#)  
     DBD/PSB/ACB Compare [187](#)  
     DBD/PSB/ACB Mapper [210](#), [216](#)  
     DBD/PSB/ACB Reversal [247](#)  
     FABMMAIN [216](#)  
     MFS Compare [390](#)  
     MFS Reversal [373](#)  
 SYSPRINT DD  
     ACBLIB Analyzer [355](#)  
     Advanced ACBGEN [336](#)  
     Consistency Checker [116](#)  
     DBD/PSB/ACB Compare [170](#)  
     DBD/PSB/ACB Mapper [203](#)  
     DBD/PSB/ACB Reversal [229](#)  
     MFS Compare [387](#)  
     MFS Reversal [368](#)  
     Multiple Resource Checker [146](#)  
 SYSPUNCH data set (DBD/PSB/ACB Reversal) [244](#)  
 SYSPUNCH DD (DBD/PSB/ACB Reversal) [229](#)  
 SYSRDDS DD (Consistency Checker) [116](#)  
 SYSUDUMP DD (Catalog Manager) [288](#)  
 SYSUT3 DD (Advanced ACBGEN) [336](#)  
 SYSUT4 DD (Advanced ACBGEN) [336](#)  
 SYSYOUT DD  
     MFS Compare [387](#)

## T

technotes [21](#)  
 time stamp checking (MFS Compare) [381](#)  
 time stamp checking (MFS Reversal) [375](#)  
 trademarks [533](#)  
 TYPE keyword (MFS Compare) [382–384](#)  
 TYPERUN=PREVUE (Advanced ACBGEN) [339](#)

## V

verification method  
     double-step verification [71](#), [80](#)  
     single-step verification [71](#), [80](#)  
 VERIFY.DB command (LICON utility) [105](#)  
 VERSID keyword (MFS Compare) [384](#)  
 VIEWLOC keyword (MFS Compare) [386](#)  
 VIEWPORT keyword (MFS Compare) [386](#)  
 VT keyword (MFS Compare) [384](#)  
 VTAB keyword (MFS Compare) [384](#)

## W

Warning Messages report (ACBLIB Analyzer) [365](#)  
 what's new [1](#)  
 WIDTH keyword (MFS Compare) [384](#)  
 WIDTH value (MFS Reversal) [379](#)  
 WINDOWF keyword (MFS Compare) [386](#)





Product Number: 5655-U08

SC19-3979-13

