

IBM IMS Fast Path Solution Pack for z/OS
2.1

Supplementary Utilities User's Guide



Note:

Before using this information and the product it supports, read the information in [“Notices” on page 247](#).

Fourth Edition (December 2023)

This edition applies to Version 2.1 of IBM IMS Fast Path Solution Pack for z/OS IMS High Performance Fast Path Utilities (program number 5698-FPP) and to any subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC27-9598-02.

© **Copyright International Business Machines Corporation 1985, 2023.**

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

Contents

About this information.....	vii
Chapter 1. Overview of IMS HP Fast Path Utilities supplementary utilities.....	1
What's new in IMS Fast Path Solution Pack Supplementary Utilities.....	2
Support for IMS-managed ACBs environment.....	3
Service updates and support information.....	3
Product documentation and updates.....	3
Accessibility features.....	5
Chapter 2. SDEP Space Utilization utility.....	7
Functions of the SDEP Space Utilization utility.....	7
Data and system flow of the SDEP Space Utilization utility.....	7
Running the SDEP Space Utilization utility.....	10
Preprocess for the SDEP Space Utilization utility: Initializing permanent data sets.....	10
Running the SDEP Space Utilization utility process.....	11
DD statements for the SDEP Space Utilization utility.....	11
FABADA7 JCL.....	12
FABADA8 JCL.....	13
DFSORT JCL (STEP SORTSDEP).....	13
FABADA9 JCL.....	14
Input for the SDEP Space Utilization utility.....	15
FABADA7 SYSIN DD data set (Standard IMS Sequential Dependent Scan Utility control statement).....	15
FABADA8 utility control statements.....	16
SORTSDEP SYSIN DD data set (DFSORT control statement).....	16
FABADA9 SYSIN DD data set.....	16
Output for the SDEP Space Utilization utility.....	17
FABADA8 SYSPRINT DD data set.....	17
FABADA9 RPTOUT DD data set.....	17
FABADA9 MSGOUT DD data set.....	19
Examples for the SDEP Space Utilization utility.....	20
Example 1: Utilizing the SDEP part.....	20
Example 2: Extracting SDEP space utilization data.....	21
Example 3: Updating only the SDEP History file.....	22
Example 4: Generating only SDEP Utilization reports.....	22
Chapter 3. Database Definition Record Create utility.....	25
Functions of the Database Definition Record Create utility.....	25
Data and system flow of the Database Definition Record Create utility.....	25
Running the Database Definition Record Create utility.....	25
DD statements for the Database Definition Record Create utility.....	26
Input for the Database Definition Record Create utility.....	28
SYSIN DD data set.....	28
Output for the Database Definition Record Create utility.....	29
SYSPRINT DD data set.....	29
REPORTS DD data set.....	29
Example for the Database Definition Record Create utility.....	33
Chapter 4. DEDB Reload Segment Data Set Create utility.....	35
Functions of the DEDB Reload Segment Data Set Create utility.....	35

Data and system flow of the DEDB Reload Segment Data Set Create utility.....	35
Calling the DEDB Reload Segment Data Set Create utility (from your program).....	36
Dynamic linkage to an application.....	36
Static linkage to an application.....	37
Application interface.....	37
DD statements for the DEDB Reload Segment Data Set Create utility.....	39
Input for the DEDB Reload Segment Data Set Create utility.....	42
UR6CTL DD data set.....	42
Output for the DEDB Reload Segment Data Set Create utility.....	46
UR6PRINT DD data set.....	46
UR6AUDIT DD data set.....	47
Setting site default values for the DEDB Reload Segment Data Set Create utility.....	48
Examples for the DEDB Reload Segment Data Set Create utility.....	50
Chapter 5. DEDB Unloaded Segment Data Set Retrieve utility.....	53
Functions of the DEDB Unloaded Segment Data Set Retrieve utility.....	53
Data and system flow of the DEDB Unloaded Segment Data Set Retrieve utility.....	53
Calling the DEDB Unloaded Segment Data Set Retrieve utility (from your program).....	54
Dynamic linkage to an application.....	54
Static linkage to an application.....	54
Application interface.....	55
DD statements for the DEDB Unloaded Segment Data Set Retrieve utility.....	58
Input for the DEDB Unloaded Segment Data Set Retrieve utility.....	61
UR7DATA, UR7DATA1, and UR7DATA2 DD data sets.....	61
UR7CTL DD data set.....	61
Output for the DEDB Unloaded Segment Data Set Retrieve utility.....	63
UR7PRINT DD data set.....	63
UR7AUDIT DD data set.....	64
Examples for the DEDB Unloaded Segment Data Set Retrieve utility.....	65
Chapter 6. HD To DEDB Unload Data Set Conversion utility.....	69
Functions of the HD To DEDB Unload Data Set Conversion utility.....	69
Data and system flow of the HD To DEDB Unload Data Set Conversion utility.....	70
Running the HD To DEDB Unload Data Set Conversion utility.....	70
DD statements for the HD To DEDB Unload Data Set Conversion utility.....	71
Input for the HD To DEDB Unload Data Set Conversion utility.....	74
UR6CTL DD data set.....	74
Output for the HD To DEDB Unload Data Set Conversion utility.....	78
UR6PRINT DD data set.....	78
UR6AUDIT DD data set.....	78
Example of the HD To DEDB Unload Data Set Conversion utility.....	79
Chapter 7. DEDB/HD Unload Conversion utility.....	81
Functions of the DEDB/HD Unload Conversion utility.....	81
Data and system flow of the DEDB/HD Unload Conversion utility.....	82
Restrictions of the DEDB/HD Unload Conversion utility.....	82
Running the DEDB/HD Unload Conversion utility.....	83
Considerations for creating databases of various formats.....	83
Creation of an HD database from a DEDB unload file.....	83
Creation of a HIDAM database from an HDAM unload file.....	84
Creation of a DEDB database from an HD unload file.....	84
Recovery and restart.....	84
PSB requirements.....	85
DD statements for the DEDB/HD Unload Conversion utility.....	85
Input for the DEDB/HD Unload Conversion utility.....	86
CNTLCRDS DD data set.....	86
SEGREFI DD data set.....	89

Segment Cross-Reference records.....	89
Output for the DEDB/HD Unload Conversion utility.....	95
SYSPRINT DD data set.....	95
Setting site default values for the DEDB/HD Unload Conversion utility.....	97
Examples for the DEDB/HD Unload Conversion utility.....	98
Example 1: Using the database definition record (DURDBDFN).....	99
Example 2: Using an HD unload file.....	100
Example 3: Using the Segment Cross-Reference table.....	101
Example 4: Segment Cross-Reference files for segment format conversions.....	107
Chapter 8. IMS DEDB randomizing module.....	111
DEDB Unload/Reload RAP number to RAP RBA conversion.....	111
Interface parameter.....	112
Link-editing your program.....	114
Required JCL DD statements to run your program.....	114
Input for the IMS DEDB randomizing module.....	115
RMIFCTL DD data set.....	115
IMS DBT 2.x application that process multiple DEDBs sequentially.....	116
Chapter 9. Standard format extract data interface module.....	117
Overview of the standard format extract data interface.....	117
Using the standard format extract data interface.....	118
Parameter list of the standard format extract data interface.....	118
Specifying JCL for the standard format extract data interface.....	119
Invoking the standard format extract data interface.....	119
Output report of standard format extract data interface.....	126
Chapter 10. Integration with IMS HP Image Copy.....	127
HPFPU Hash Check support for IMS HP Image Copy.....	127
Input for HPFPU Hash Check support for IMS HP Image Copy.....	127
Output for HPFPU Hash Check support for IMS HP Image Copy.....	127
Examples: HPFPU Hash Check support for IMS HP Image Copy.....	131
DB Sensor support for IMS HP Image Copy.....	132
Input for DB Sensor support for IMS HP Image Copy.....	132
Output for DB Sensor support for IMS HP Image Copy.....	132
Example: DB Sensor support for IMS HP Image Copy.....	133
Chapter 11. Integration with IMS Database Recovery Facility.....	135
HPFPU Hash Check support for IMS Database Recovery Facility.....	135
Input for HPFPU Hash Check for IMS Database Recovery Facility.....	135
Output for HPFPU Hash Check support for IMS Database Recovery Facility.....	135
FPA Build Index support for IMS Database Recovery Facility.....	136
Input for FPA Build Index support for IMS Database Recovery Facility.....	136
Output for FPA Build Index support for IMS Database Recovery Facility.....	136
Chapter 12. Reference: Supplementary utility reports stored in IMS Tools KB.....	137
Chapter 13. Troubleshooting.....	139
Messages.....	139
FABA messages.....	140
FABC messages.....	149
FABD messages.....	230
HFPB messages.....	239
Gathering diagnostic information.....	246
Notices.....	247

Index.....	251
-------------------	------------

About this information

IBM® IMS Fast Path Solution Pack for z/OS® IMS High Performance Fast Path Utilities (also referred to as IMS HP Fast Path Utilities) improves performance and availability by streamlining database administrator (DBA) tasks.

To use the procedures in this information, you must first install IMS HP Fast Path Utilities as described in the *Program Directory for IMS Fast Path Solution Pack for z/OS, 2.1, GI13-5905*, and then perform the post-installation steps as described in the *IMS Fast Path Solution Pack: Overview and Customization, GC27-9596*.

These topics are designed for database administrators and technical support personnel who are involved in database management, maintenance, and performance tuning, and require a knowledge of how to operate the supplementary utilities of IMS HP Fast Path Utilities, and are specifically for those who manage the IMS Data Entry Databases (DEDBs). These topics help database administrators and technical support personnel to perform these tasks:

- Understand the functions of IMS HP Fast Path Utilities supplementary utilities
- Run and use IMS HP Fast Path Utilities supplementary utilities after they are installed
- Use DD statements to control how you use IMS HP Fast Path Utilities supplementary utilities

IMS HP Fast Path Utilities includes all the features you need to manage your IMS Fast Path databases.

For information about other utilities and tools of IMS Fast Path Solution Pack IMS HP Fast Path Utilities, see the following information:

- *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*
- *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*

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

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

Always check the IMS Tools Product Documentation page for complete product documentation resources:

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

The IMS Tools Product Documentation page includes:

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

Chapter 1. Overview of IMS HP Fast Path Utilities supplementary utilities

IBM IMS Fast Path Solution Pack for z/OS IMS High Performance Fast Path Utilities (also referred to as IMS HP Fast Path Utilities) provides many supplementary utilities to help you manage IMS Fast Path databases more efficiently.

The following table summarizes all the supplementary utilities that are included in IMS HP Fast Path Utilities.

Table 1. Supplementary utilities of IMS HP Fast Path Utilities

Utility or program name	Function
SDEP Space Utilization utility	Generates reports that show trends in SDEP space utilization. You can use the information provided to schedule database expansion and to forecast future DASD requirements, thus utilize the SDEP.
Database Definition Record Create utility (FABCUR5)	Creates the database definition record data set (DURDBDFN), which can be used as one of the inputs to FABCUR3, FABCUR7, and FABCUR9.
DEDB Reload Segment Data Set Create utility (FABCUR6)	Enables a user application program to create a DEDB reload segment data set, which can be used as one of the inputs to the FPA Reload function, FABCUR3, FABCUR7, and FABCUR9.
DEDB Unload Segment Data Set Retrieve utility (FABCUR7)	Enables a user application program to retrieve unloaded DEDB database segments from the DEDB reload segment data set in hierarchical order.
HD To DEDB Unload Data Set Conversion utility (FABCUR8)	Converts an HD unload data set to a DEDB Unloaded segment data set.
DEDB Unload Conversion utility (FABCUR9)	Loads data from various formats of unload files onto an IMS full-function or Fast Path DEDB database.
IMS DEDB Randomizing module (FABCRMIF)	Enables an application program to invoke a DEDB randomizer.
Standard format extract data interface module (FPXGXDR0)	Insulates user-written application programs from future changes to the standard format of the extract function.
HPFPU Hash Check support for IMS HP Image Copy	Invokes the HPFPU HASH Check support during IMS HP Image Copy jobs.
DB Sensor support for IMS HP Image Copy	Collects sensor data within IMS HP Image Copy jobs.
HPFPU Hash Check support for IMS Database Recovery Facility	Invokes the HPFPU HASH Check support during IMS Database Recovery Facility jobs.
FPA Build Index support for IMS Database Recovery Facility	Builds secondary index databases in IMS Database Recovery Facility jobs.

Topics:

- [“What's new in IMS Fast Path Solution Pack Supplementary Utilities” on page 2](#)
- [“Support for IMS-managed ACBs environment” on page 3](#)
- [“Service updates and support information” on page 3](#)

- [“Product documentation and updates” on page 3](#)
- [“Accessibility features” on page 5](#)

What's new in IMS Fast Path Solution Pack Supplementary 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.

SC27-9598-03 (December 2023)

Description	Related APARs
HPFPU Hash Check support for IMS Database Recovery Facility: IMS managed ACBs environment support. The following topic is updated: “Support for IMS-managed ACBs environment” on page 3 .	PH57462

SC27-9598-02 (January 2023)

Description	Related APARs
HPFPU Hash Check support and DB Sensor support for IMS HP Image Copy: IMS-managed ACBs environment support. The following topic is updated: “Support for IMS-managed ACBs environment” on page 3 .	PH50776

SC27-9598-01 (January 2022)

Description	Related APARs
HPFPU Hash Check support for IMS HP Image Copy can generate historical records in the HFP AHST data set. The following topics are added or modified: <ul style="list-style-type: none"> • “HFP AHST DD data set” on page 130 • “Examples: HPFPU Hash Check support for IMS HP Image Copy” on page 131 • New message: FABA4040E 	PH28270
Modified message: HFPB0005I	N/A

Support for IMS-managed ACBs environment

In an IMS managed ACBs environment, IMS can manage the runtime application control blocks (ACBs) for databases and program views for you. When IMS manages ACBs, IMS no longer requires DBD, PSB, and ACB libraries.

The following supplementary utilities support IMS-managed ACBs environment:

- DEDB Reload Segment Data Set Create utility (FABCUR6). Requires the IMSCATHLQ statement in IMS-managed ACBs environment.
- DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7). Requires the IMSCATHLQ statement in IMS-managed ACBs environment.
- IMS DEDB randomizing module interface module (FABCRMIF or FABDRMIF). Requires the IMSCATHLQ statement in IMS-managed ACBs environment.
- Integration with IMS HP Image Copy
 - HPFPU Hash Check support for IMS HP Image Copy
 - DB Sensor support for IMS HP Image Copy

For more information, see the topic "Considerations when the IMS management of ACBs is enabled" in the *IMS High Performance Image Copy User's Guide*.

- Integration with IMS Database Recovery Facility
 - HPFPU Hash Check support for IMS Database Recovery Facility
 - FPA Build Index support for IMS Database Recovery Facility

For more information, see the topic "IMS-managed ACBs environment restrictions" in the *IMS Recovery Solution Pack IMS Database Recovery Facility User's Guide*.

Supplementary utilities use the IMS Tools Catalog Interface to process the IMS catalog directory. To learn more about the interface, see the topic "IMS Tools Catalog Interface" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

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:

[IBM Support: IMS Fast Path Solution Pack for z/OS](#)

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 automated 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, you can take one of the following actions:

- Click the Feedback button at the top of the IBM Documentation topic that you are commenting on.
- Click the Contact Us tab at the bottom of any IBM Documentation topic.
- Send an email to ibmdocs@us.ibm.com. Be sure to include the book title, topic or section title, specific text, and your comment.

To help us respond quickly and accurately, 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 and publications

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

The IMS publications are prerequisite for all IMS HP Fast Path Utilities components.

Related publications

This information describes supplementary utilities of IMS HP Fast Path Utilities. For information about other utilities and tools of IMS Fast Path Solution Pack IMS HP Fast Path Utilities, see the following information:

- *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide (SC27-9536)*
- *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide (SC27-9597)*

This information refers to information in other guides using shortened versions of the information titles. The following table contains a list of information referred to by their short titles:

Short title used in this information	Title	Order number
<i>IMS Fast Path Solution Pack: Overview and Customization</i>	<i>IBM IMS Fast Path Solution Pack for z/OS 2.1: Overview and Customization</i>	GC27-9596
<i>IMS High Performance Image Copy User's Guide</i>	<i>IBM IMS High Performance Image Copy for z/OS 4.2 User's Guide</i>	SC19-2756
<i>IMS Recovery Solution Pack IMS Database Recovery Facility User's Guide</i>	<i>IBM IMS Recovery Solution Pack for z/OS 2.1 IMS Database Recovery Facility User's Guide</i>	SC27-8441
<i>IMS Solution Packs Data Sensor User's Guide</i>	<i>IBM IMS Solution Packs Data Sensor User's Guide</i>	SC19-3283
<i>IMS Tools Base Configuration Guide</i>	<i>IBM IMS Tools Base for z/OS Configuration Guide</i>	SC19-4370 SC27-9852
<i>IMS Tools Base IMS Tools Knowledge Base User's Guide</i>	<i>IBM IMS Tools Base for z/OS IMS Tools Knowledge Base User's Guide</i>	SC19-4372 SC27-9855

Accessibility features

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

Accessibility features

The major accessibility feature in IMS HP Fast Path 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, refer to the following publications

- *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 for more information about the commitment that IBM has to accessibility.

Chapter 2. SDEP Space Utilization utility

Use the SDEP Space Utilization utility to generate reports that show trends in SDEP space utilization. You can use the information provided to schedule database expansion and to forecast future DASD requirements, thus utilize the SDEP.

Topics:

- [“Functions of the SDEP Space Utilization utility” on page 7](#)
- [“Data and system flow of the SDEP Space Utilization utility” on page 7](#)
- [“Running the SDEP Space Utilization utility” on page 10](#)
- [“DD statements for the SDEP Space Utilization utility” on page 11](#)
- [“Input for the SDEP Space Utilization utility” on page 15](#)
- [“Output for the SDEP Space Utilization utility” on page 17](#)
- [“Examples for the SDEP Space Utilization utility” on page 20](#)

Functions of the SDEP Space Utilization utility

SDEP space utilization data is extracted and written to an OS file by a routine that runs as a Fast Path online utility.

Specifically, it is an exit routine that is invoked by the IMS DEDB Sequential Dependent Scan Utility (DBFUMSC0). The extraction routine was designed to cause virtually no database access contention with other Fast Path or mixed mode applications. The routine accesses and analyzes the in-storage DMAC control block, and writes a data record to a file defined by the SCANCOPY DD statement. During each invocation of the IMS DEDB Sequential Dependent Scan Utility, all or specified areas of a DEDB can be analyzed.

Most applications that use sequential dependent segments run the IMS DEDB Sequential Dependent Scan Utility to copy the sequential dependent segments to a sequential data set, and then by the DEDB Sequential Dependent Delete Utility (DBFUMDL0) to logically delete the segments. The DEDB Pointer Checker extraction routine is intended to be run just before the delete utility so that the "high-water" space utilization mark can be captured. However, this routine can be run as often as desired, and the space utilization graph will show the largest amount of space used during each 24-hour period.

Other SDEP Space Utilization utility batch programs use the extracted data to update the SDEP History file and to generate the SDEP Space Utilization reports. The SDEP History file is a VSAM KSDS with one record per DEDB area.

SDEP Space Utilization report

This report presents a rolling 30-day graph that depicts space utilization of total dependent part. Indicators are highlighted when the SDEP part is "wrapped" and when physical changes are made to the area. The physical and organizational characteristics and the size of the area are shown.

This report shows trends in space utilization. It provides the information required to schedule database expansion and to forecast future DASD requirements.

Data and system flow of the SDEP Space Utilization utility

This topic describes the data and system flow of the SDEP Space Utilization utility.

The following figure shows the general data flow for the SDEP Space Utilization utility. Input consists of the DEDB area and the ACBLIB data sets, which are controlled by the IMS online system, and the SYSIN data set. The SDEP space utilization data, that is, the data between the SDEP logical beginning and the logical end in the area are gathered by the exit routine of the IMS DEDB Sequential Dependent Scan

utility. Output consists of the job log messages, the SDEP Space Utilization report, and the SDEP history file.

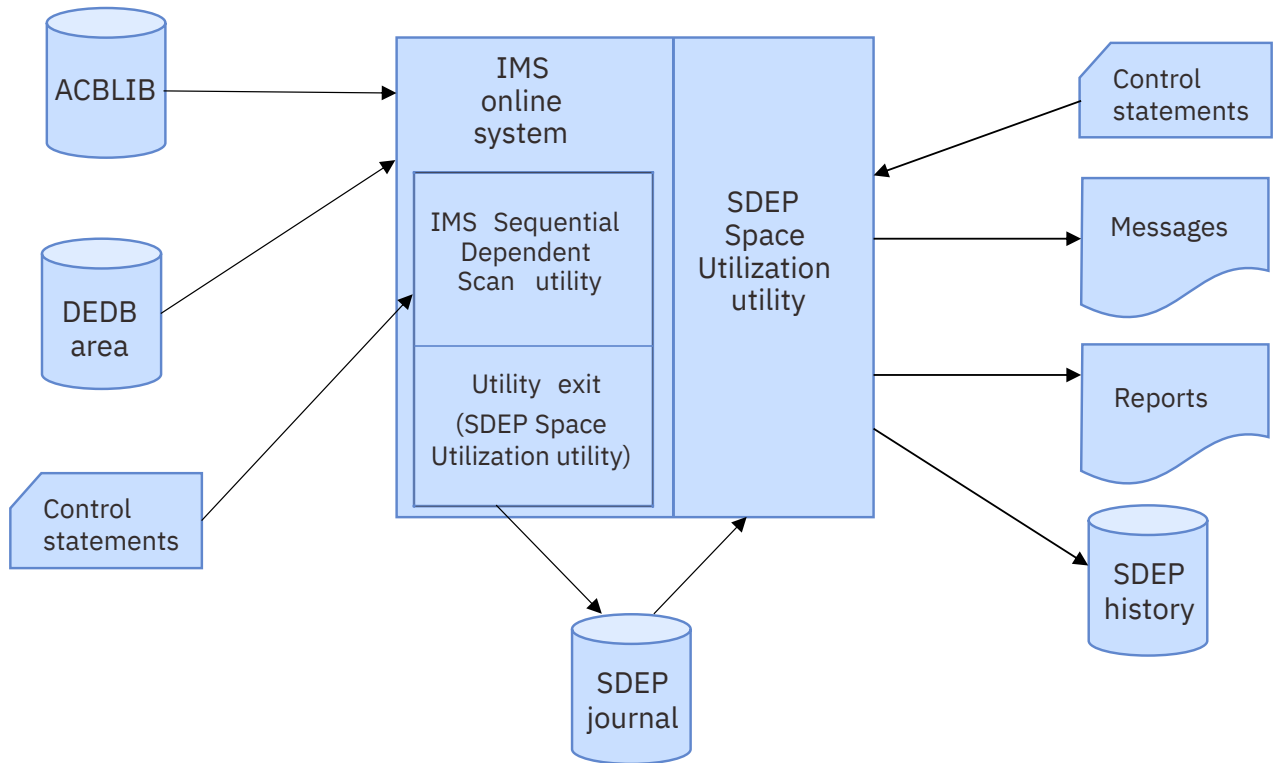


Figure 1. Data and system flow of the SDEP Space Utilization utility

Subsections:

- [“Processing flow of the SDEP Space Utilization utility” on page 8](#)
- [“Load modules of the SDEP Space Utilization utility” on page 10](#)

Processing flow of the SDEP Space Utilization utility

There are four processes in the SDEP Space Utilization utility: FABADA7, FABADA8, DFSORT, and FABADA9.

The addressing mode of FABADA7 depends on the IMS level under which it is invoked. All other programs run in 24-bit addressing mode.

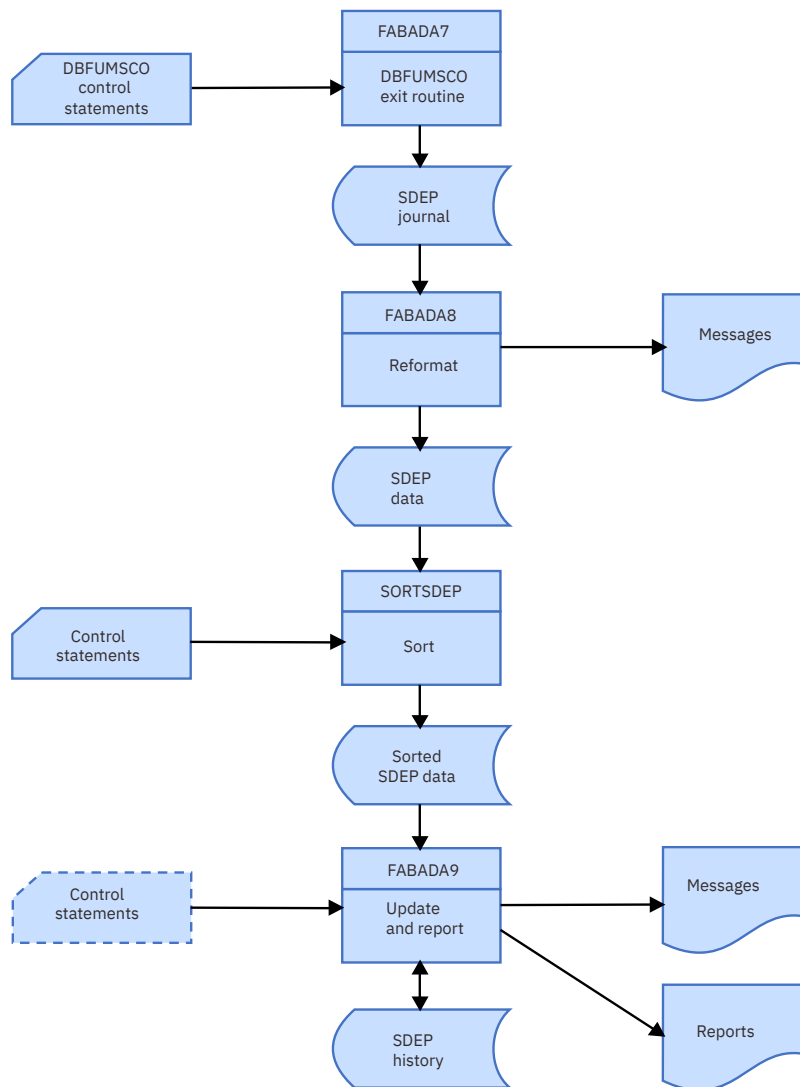


Figure 2. Processing flow of the SDEP Space Utilization utility

The SDEP part processing steps include:

FABADA7

This program extracts the SDEP space utilization data and writes it to an SDEP Data Collection file. This module runs as a Fast Path online utility. Specifically, it is an exit routine that is invoked by the IMS Sequential Dependent Scan utility. It accesses and analyzes the in-storage DMAC.

FABADA8

This program copies and re-formats the SDEP space utilization records (created by FABADA7) and resets the end-of-file pointer on the SDEP Data Collection file.

DFSORT

The IBM Data Facility Sort sorts the DADARO data set that is created by FABADA8.

FABADA9

This program updates the SDEP History file using the records that were reformatted by FABADA8, and generates the SDEP Utilization reports for specified databases or areas. This program can run in update only mode, report only mode, or combined mode. If run in update only mode, a utilization report is automatically generated if the utilization threshold is exceeded.

The steps vary depending on your particular functional options. A typical job contains some or all of the processing steps described here.

Load modules of the SDEP Space Utilization utility

The SDEP Space Utilization utility contains three load modules: FABADA7, FABADA8, and FABADA9. The following table lists these modules.

Table 2. Load modules of the SDEP Space Utilization utility

Load module name	Function
FABADA7	Extract SDEP space utilization data
FABADA8	Extract required information
FABADA9	Update SDEP History file and generate reports

Running the SDEP Space Utilization utility

There are two main steps in running the SDEP Space Utilization utility.

1. Allocate and initialize all permanent data sets required by the SDEP Space Utilization utility. This is a preprocess to run the SDEP Space Utilization utility, and is done only once.
2. Run the SDEP Space Utilization utility.

Preprocess for the SDEP Space Utilization utility: Initializing permanent data sets

To use the SDEP Space Utilization utility, you must have the SDEP Data Collection data set and the SDEP History data set initialized.

Table 3. SDEP Data Collection data set and SDEP History data set

SDEP Data Collection data set	SDEP History data set
<p>This data set is used by modules FABADA7 and FABADA8. Whenever you run FABADA7, this data set is defined by the SCANCOPY DD statement. You must use DISP=(MOD,KEEP,KEEP). When you run FABADA8, this data set is defined by the DADARI DD statement. You must use: DISP=(OLD,KEEP,KEEP).</p> <p>Each time you run FABADA7, one record is written to this data set. When FABADA8 is run, it reads and processes all records on the SDEP Data Collection data set. Then it removes all records, leaving the data set empty.</p>	<p>This data set is used by module FABADA9. It is defined by the DASDHIO DD statement. You must use DISP=OLD.</p> <p>Each time you run FABADA9 (preceded by FABADA8 and DFSORT), the SDEP History data set is updated with the data that was collected by FABADA7, reformatted by FABADA8, and sorted by DFSORT. FABADA9 maintains a 30-day history of each DEDB area that was processed by FABADA7.</p>

The following figure shows the JCL that you should use to allocate the SDEP Data Collection data set and the SDEP History data set. The initialization of this KSDS with two records (low-values key and high-values key) is required.

```

//*****
//** ALLOCATE AND INITIALIZE SDEP HISTORY DATA SET
//*****
//IEBDG EXEC PGM=IEBDG
//SYSPRINT DD SYSOUT=A
//INITFYL DD DSN=&INITFYL,DISP=(,PASS),
//          UNIT=SYSDA,SPACE=(TRK,(1)),
//          DCB=(RECFM=FB,LRECL=800,BLKSIZE=800)
//SYSIN DD *
DSD OUTPUT=(INITFYL)
CREATE QUANTITY=1, X
      FILL=X'00', X
      PICTURE=37,17,'IMS SYSTEM UTILITIES/DATA BASE TOOLS '
CREATE QUANTITY=1, X
      FILL=X'FF', X
      PICTURE=37,17,'IMS SYSTEM UTILITIES/DATA BASE TOOLS ' END
/*
//*
//IDCAMS EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//INITFYL DD DSN=&INITFYL,DISP=(OLD,PASS)
//SYSIN DD *
DELETE ( HPFP.SDEP.HIST ) CLUSTER
DELETE ( HPFP.SDEP.DATA ) NONVSAM
SET MAXCC = 0
DEFINE CLUSTER ( NAME(HPFP.SDEP.HIST) -
                  VOLUMES(NNNNNN) -
                  INDEXED UNIQUE -
                  RECORDSIZE(800 800) -
                  KEYS (16 00) -
                  CISZ(4096) -
                  RECORDS(100) ) -
DATA ( NAME(HPFP.SDEP.HIST.DATA) ) -
INDEX ( NAME(HPFP.SDEP.HIST.INDX) )
IF MAXCC < 8 THEN -
  REPRO INFILE(INITFYL) ODS(HPFP.SDEP.HIST)
/*
//*****
//** ALLOCATE SDEP DATA COLLECTION DATA SET
//*****
//IEFBR14 EXEC PGM=IEFBR14
//SCANCOPY DD DSN=HPFP.SDEP.DATA,DISP=(NEW,CATLG,CATLG),
//          UNIT=SYSDA,SPACE=(TRK,(7,2),RLSE)
//*

```

Figure 3. Sample JCL to initialize permanent data sets: SDEP Data Collection and SDEP History data sets

Running the SDEP Space Utilization utility process

To use the SDEP Space Utilization utility, you must run several programs. To run the programs, you must code JCL statements for those programs.

Procedure

1. Code the JCL for the SDEP Space Utilization utility (FABADA7, FABADA8, and FABADA9) and DFSORT job steps that you need to run.
2. Code the control statements needed for the programs.
3. Make a test run. (FABADA7 must be run as the FP Online utility exit routine.)
4. Interpret the output reports to verify that process completed successfully.
5. Put the resulting JCL and control-statement into production use.

DD statements for the SDEP Space Utilization utility

DD statements for the SDEP Space Utilization utility determine the input and output data sets and how the SDEP Space Utilization utility is run.

You must specify DD statements for the job control language (JCL) for each of the SDEP Space Utilization utility programs.

FABADA7 JCL

The FABADA7 program is used to extract and journal SDEP space utilization data.

The data is extracted and written to an OS file by a routine that runs as a Fast Path online utility. Specifically, FABADA7 is an exit routine that is invoked by the DEDB Sequential Dependent Scan utility (DBFUMSC0). It accesses and analyzes the in-storage DMAC and writes a data record to a MOD file defined by the SCANCOPY DD statement.

During each invocation of the DBFUMSC0, space utilization data can be extracted from all or specified areas of a DEDB.

Warning: When you run DBFUMSC0 with the FABADA7 exit routine, no SDEP segments are written in the SCANCOPY file. This means that the FABADA7 run must be a different job step from your normal DBFUMSC0 step.

Most application systems that use sequential dependents run DBFUMSC0 to copy the sequential dependent segments to a sequential data set, followed by the IMS Sequential Dependent Delete utility (DBFUMDL0) to logically delete the segments.

The extraction routine is intended to be run just before the delete utility so that the "high-water" space utilization mark can be captured. However, this utility can be run as often as desired and the Space Utilization graph always reflects the highest amount of space used in each 24-hour period.

Standard DBFUMSC0 JCL, as described in *IMS Database Utilities*, must be used. You should use the FPUTIL cataloged procedure (see *IMS System Definition*), providing an EXEC statement and DD statements as follows:

EXEC

This statement must be in the form:

```
// EXEC FPUTIL, DBD=dbdname, REST=00, RGN=256K
```

FPU.STEPLIB DD

You must concatenate the partitioned data set that contains the FABADA7 load module to the procedure STEPLIB. This statement should be in the following form:

```
//FPU.STEPLIB DD  
// DD DSN=HPFP.SHFPLMD0, DISP=SHR
```

SCANCOPY DD

This is the SDEP data collection data set. It should be preallocated. You must use DISP=(MOD,KEEP,KEEP).

```
//SCANCOPY DD DSN=HPFP.SDEP.DATA, DISP=(MOD,KEEP,KEEP)
```

Only SDEP space utilization data records are written in this data set. No SDEP segments are written.

SYSIN DD

Defines the input control statement data set. This data set can reside on a direct-access device or be routed through the input stream. It contains standard DBFUMSC0 control statements. For information, see *IMS Database Utilities*.

Your first two control statements must be:

```
TYPE SCAN  
ERROR SCAN
```

Then, for each area you want to process, you must include these statements:

```
AREA areaname  
EXIT FABADA7  
GO
```

Related reference

Preprocess for the SDEP Space Utilization utility: Initializing permanent data sets

To use the SDEP Space Utilization utility, you must have the SDEP Data Collection data set and the SDEP History data set initialized.

Example 2: Extracting SDEP space utilization data

These are example JCL statements for extracting SDEP space utilization data.

FABADA8 JCL

The FABADA8 program, in conjunction with FABADA9, is used to update the SDEP History file using the data records journaled by FABADA7, and to generate SDEP Space Utilization reports for all or specified areas of any database.

This program re-formats the SDEP space utilization data records, and resets the end-of-file marker on the journal data set.

There is no control statement needed for running FABADA8.

FABADA8 is run as a standard z/OS job step. An EXEC statement and DD statements that define inputs and outputs are required. The following table summarizes the DD statements.

Table 4. FABADA8 DD statements

DDNAME	Use	Format	Required or optional
DADARI	Input/Output	DISP=OLD	Required
SYSPRINT	Output	LRECL=133	Required
DADARO	Output	LRECL=80	Required

EXEC

This statement must be in the form:

```
// EXEC PGM=FABADA8,REGION=512K
```

DADARI DD

This statement defines the input and output data set that contains the SDEP space utilization data records. This DD statement should always specify DISP=OLD. This is the SCANCOPY data set from FABADA7.

SYSPRINT DD

This statement defines the output message data set. The data set can reside on tape, direct-access device or printer, or be routed through the output stream. You should code your DD statement as follows:

```
//SYSPRINT DD SYSOUT=A
```

DADARO DD

This statement defines the output data set that contains the reformatted SDEP space utilization data records. Block size must be a multiple of 80. Do not specify DISP=MOD for this DD statement.

DFSORT JCL (STEP SORTSDEP)

The DFSORT program sorts the DADARO data set that is created by FABADA8. Sorted data set is used by FABADA9.

To run DFSORT, you have to supply the appropriate DD statements. The following table summarizes the DD statements needed to run DFSORT. All statements in this table are required.

Table 5. DFSORT DD statements (STEP SORTSDEP)

DDNAME	Use	Format	Required or optional
SORTIN	Input		Required
SYSIN	Input		Required
SORTOUT	Output		Required
SYSOUT	Output	SYSOUT	Required
SORTWK01	Work data set		Required
SORTWK02	Work data set		Required
SORTWK03	Work data set		Required

EXEC

This statement must be in the following form:

```
// EXEC PGM=SORT
```

SORTIN DD

This input data set is the DADARO file from FABADA8.

SYSIN DD

This input data set contains DFSORT control statements. Code this as follows:

```
//SYSIN DD *
  SORT FIELDS=(1,24,CH,A)
  END
/*
```

SORTOUT DD

This output data set contains the sorted records. It is used by FABADA9. Required space is the same size as the SORTIN data set. Do not specify DISP=MOD for this DD statement.

SYSOUT DD

This output data set contains the message produced by DFSORT.

SORTWKnn DD

These are intermediate storage data sets used by DFSORT. See *DFSORT Application Programming Guide* for more information on how to code SORTWKnn DD statements.

Allocating twice the space used by the SORTIN data set is usually adequate for each work data set.

FABADA9 JCL

This program uses the SDEP space utilization data records (sorted) to update the SDEP History file and generates Space Utilization reports for all or specified areas of any number of databases.

FABADA9 is run as a standard z/OS job step. An EXEC statement and DD statements that define inputs and outputs are required. The following table summarizes the DD statements needed to run DFSORT. All statements in this table are required.

Table 6. FABADA9 DD statements

DDNAME	Use	Format	Required or optional
SYSIN	Input	LRECL=80	Required
DADARI	Input		Required
MSGOUT	Output	LRECL=133	Required
RPTOUT	Output	LRECL=133	Required

Table 6. FABADA9 DD statements (continued)

DDNAME	Use	Format	Required or optional
DASDHIO	Input/Output	KSDS	Required

EXEC

This statement must be in the form:

```
// EXEC PGM=FABADA9,REGION=768K,PARM='THR=nn'
```

THR=nn on the EXEC statement PARM parameter specifies the threshold space utilization value. Valid values are 00 - 99. If omitted, the default is 85. If the space utilization exceeds the threshold value, a special return code is set (99) and a report is generated even when update only is specified.

SYSIN DD

This statement defines the input control statement data set. This data set can reside on tape, a direct-access device, or be routed through the input stream.

DADARI DD

This statement defines the (sorted) input data set that contains the SDEP space utilization data records. This is the SORTOUT data set from SORTSDEP.

MSGOUT DD

This statement defines the output message data set. The data set can reside on tape, a direct-access device or printer, or be routed through the output stream. You should code your DD statement as follows:

```
//MSGOUT DD SYSOUT=A
```

RPTOUT DD

This statement defines the output SDEP utilization report data set. The data set can reside on tape, a direct-access device or printer, or be routed through the output stream. You should code your DD statement as follows:

```
//RPTOUT DD SYSOUT=A
```

DASDHIO DD

This statement defines the SDEP History file (VSAM KSDS). This data set must be allocated and initialized before you start running FABADA9. You must use DISP=OLD for this data set.

For the details, see [Table 3 on page 10](#) and [“Preprocess for the SDEP Space Utilization utility: Initializing permanent data sets” on page 10.](#)

Input for the SDEP Space Utilization utility

To run the SDEP Space Utilization utility, you must specify input definitions, including the various control statements, for each program.

FABADA7 SYSIN DD data set (Standard IMS Sequential Dependent Scan Utility control statement)

Standard IMS Sequential Dependent Scan utility commands are used.

You should use only the following command keywords:

```
TYPE
ERROR
AREA
EXIT
BUFNO
FIXOPT
```

```
NOSORT  
GO
```

Related reference

[Example 2: Extracting SDEP space utilization data](#)

These are example JCL statements for extracting SDEP space utilization data.

FABADA8 utility control statements

The FABADA8 program does not require control statements.

SORTSDEP SYSIN DD data set (DFSORT control statement)

You need to add one line of control statement for the SORTSDEP SYSIN DD data set.

The required control statement is the following:

```
SORT FIELDS=(1,24,CH,A)
```

FABADA9 SYSIN DD data set

The SYSIN DD data set contains the user's description of the processing to be done by module FABADA9. It describes the database and area for which a utilization graph is to be generated.

Control statement syntax

The syntax of the control statement discussed here is applicable to FABADA9 control statement.

FABADA9 requires a control statement.

Keywords and the associated values can be coded in free format (columns 1 - 72), provided certain syntactical coding rules are followed:

1. The keyword and its value must be on one control statement.
2. If specification of a keyword value is required, the keyword must be separated from its associated value by an equal sign. The equal sign must not be preceded by blanks, but can be followed by one or more blanks. The value must be separated from the next keyword by a blank, a comma, or a comma followed by one or more blanks.
3. For keywords that do not have associated values, the keyword must be separated from the next keyword by a blank, a comma, or a comma followed by one or more blanks.
4. In case of duplicate keywords, the last one coded is used.

Format

This control-statement data set usually resides in the input stream. However, it can also be defined as a sequential data set or as a member of a partitioned data set. It must contain 80-byte fixed-length records. Block size, if coded, must be a multiple of 80.

This data set must contain only one control statement. It can be coded as shown in the following figure:

```
//FABADA9.SYSIN DD *  
  DBDNAME=VRSDSRF,AREA=VRSTSS1  
/*
```

Figure 4. FABADA9 SYSIN DD data set

Record format

There is only one statement type in the SYSIN file. It contains the following keywords:


```
DBDNAME=dbdname  
[AREA=areaname]
```

DBDNAME=

This keyword specifies the name of the database for which SDEP Space Utilization reports are to be generated. If an AREA= keyword is not present, reports are produced for all areas that are included in SDEP History data set. DBDNAME is a required keyword.

AREA=

This keyword specifies the DDNAME of the specific area for which an SDEP Space Utilization report is to be generated. AREA is an optional keyword.

Related reference

[Example 1: Utilizing the SDEP part](#)

The following figure shows example JCL statement for utilizing the SDEP part.

Output for the SDEP Space Utilization utility

The following topics describe the output (that is, reports) that are generated by the SDEP Space Utilization utility, their formats and data elements, and how to get and interpret them.

FABADA8 SYSPRINT DD data set

The SYSPRINT DD data set contains the messages issued by the FABADA8 program.

Format

This data set contains 133-byte records, and block size (if coded in your JCL statement) must be a multiple of 133. You should code your DD statement as follows:

```
//SYSPRINT DD SYSOUT=A
```

SDEP Data Format-Message

The following figure shows an example of the SDEP Data Format-Messages.

```
IMS HFPF UTILITIES - DEDBPC                "SDEP DATA FORMAT - MESSAGES"                PAGE: 1  
5698-FPP                                DATE: 11/22/2020  TIME: 20.13.24          FABADA8 - V2R1  
  
FABA0800I - FABADA8 ENDED NORMALLY  
NUMBER OF INPUT RECORDS READ : 3
```

Figure 5. SDEP Data Format-Messages

FABADA9 RPTOUT DD data set

The RPTOUT DD data set contains the SDEP Utilization report produced by the FABADA9 program.

Format

This data set contains 133-byte fixed-length records, and block size (if coded in your JCL statement) must be a multiple of 133. You should code your DD statement as follows:

```
//RPTOUT DD SYSOUT=A
```

SDEP Utilization report

Purpose

The SDEP Utilization report provides the following:

- A graph showing the amount of SDEP space used on each of the last 31 days
- Indicators to identify when the SDEP area "wrapped," the DBD changed or the area was initialized
- The attributes of the area.

Report content

The following figure shows an example of the SDEP Utilization report.

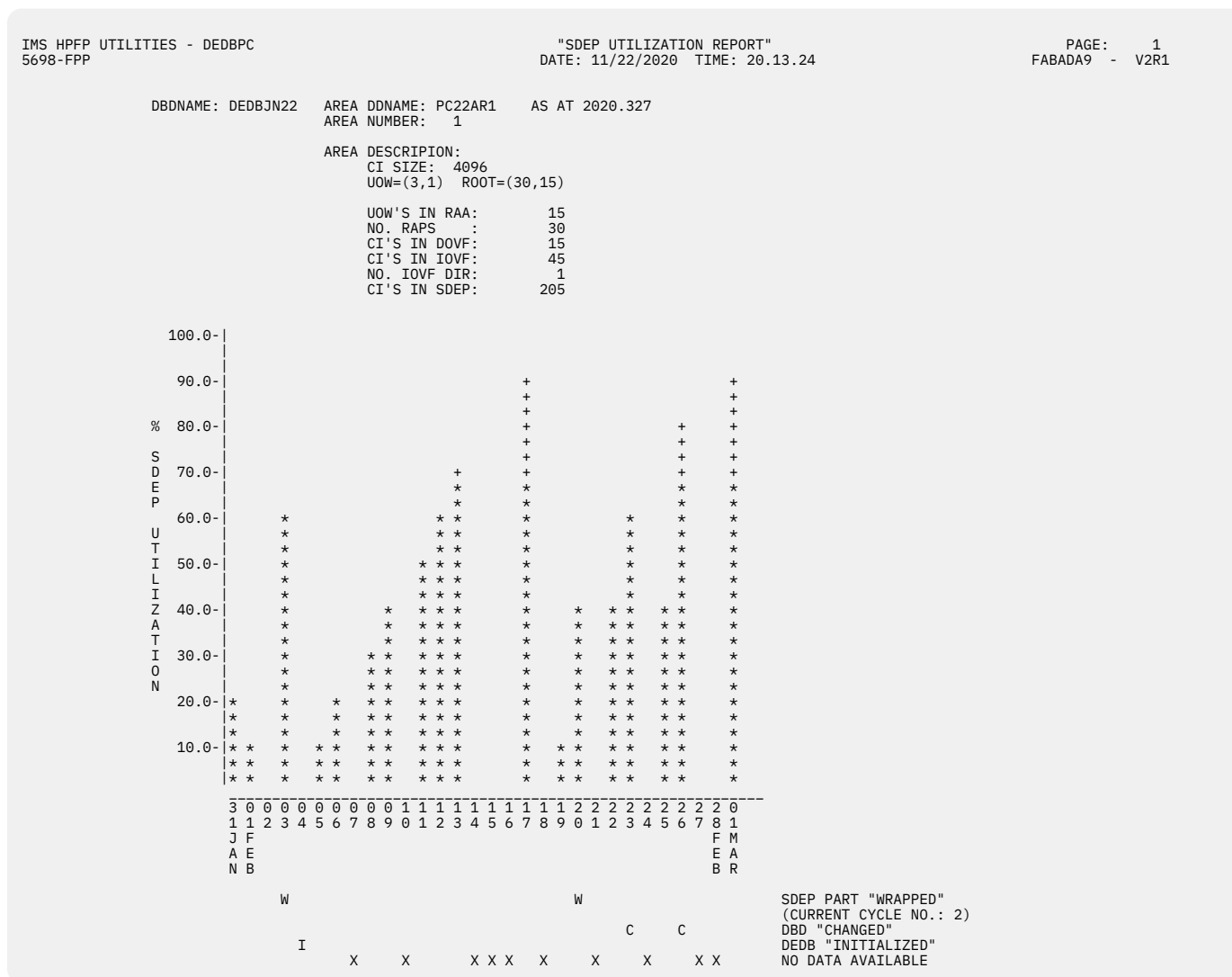


Figure 6. SDEP Utilization report

The following are the descriptions for this report:

DATE: mm/dd/yyyy

TIME: hh:mm:ss

These are shown in the report header. The date and time of the report generation or update run.

AS At yyyy.ddd

The Julian date of the last History file entry for this entry

<<THRESHOLD EXCEPTION DETECTED>>

Appears if the report was generated because of a threshold violation.

AREA DESCRIPTION

Shows the attributes and size of the area.

% SDEP UTILIZATION (the vertical axis)

The percentage of SDEP area used in increments of 3.3. The utilization calculation always rounds up to the next higher increment (that is, any calculated percentage between 10.1 and 13.3 appears on the graph as 13.3%).

Each graph line shows the total SDEP area utilization for the day specified by the horizontal axis. A plus sign (+) shows that the space utilization threshold value has been exceeded. Default threshold value is 85%.

SDEP PART "WRAPPED"

A wrap indicator (W) on this line shows when the area "wrapped" (that is, the cycle counter incremented). This line does not appear on the report if the SDEP section did not wrap during the reporting period. The current cycle counter is displayed as part of the message.

DBD "CHANGED"

A change indicator (C) on this line identifies when the size of the VSAM cluster was changed, or changes were made to one or more of the following DBD parameters. This line does not appear on the report if no changes were made during the reporting period.

- DEVICE TYPE
- CI SIZE
- ROOT values
- UOW values.

DEDB "INITIALIZED"

An initialization indicator (I) on this line identifies when the VSAM cluster was initialized (that is, the DEDB Initialization utility DBFUMINO was run). This line does not appear on the report if the area was not initialized during the reporting period.

NO DATA AVAILABLE

A no data indicator (X) on this line identifies the days for which no history data is available (that is, the online utility that extracts SDEP utilization information was not run for this area).

Usage

The SDEP Utilization report is an aid for monitoring the space utilization of the sequential dependent part of a database.

Rules of Thumb:

- When the peak utilization trend line nears 90%, expansion of the database may be required.
- Peak utilization below 60% implies that excessive space has been allocated for the data set.
- This report should be reviewed at least weekly for regular production systems, and more frequently for new systems.
- A graph of an area is automatically generated if the space utilization exceeds a user-specified threshold value.

FABADA9 MSGOUT DD data set

The MSGOUT DD data set contains the messages issued by the FABADA9 program. These include all messages that report integrity problems with your DEDB area.

Format

This data set contains 133-byte records, and block size (if coded in your JCL statement) must be a multiple of 133. You should code your DD statement as follows:

```
//SYSPRINT DD SYSOUT=A
```

SDEP History/Reports—Messages

The following figure shows an example of the SDEP History/Reports Messages.

```
IMS HFPF UTILITIES - DEDBPC          "SDEP HISTORY/REPORTS - MESSAGES"
5698-FPP                           DATE: 11/22/2020  TIME: 20.13.24
                                     PAGE:      1
                                     FABADA9 - V2R1

FABA0911I - UPDATE PHASE ENDED NORMALLY
FABA0950I - CARD   1 :   DBDNAME=DEDBJN22,AREA=PC22AR1
FABA0931I - REPORT PHASE ENDED NORMALLY
FABA0900I - FABADA9 ENDED NORMALLY
```

Figure 7. SDEP History/Reports—Messages

Examples for the SDEP Space Utilization utility

There are many ways to run the SDEP Space Utilization utility. The examples provided in the following topics show some of the typical ways that you can use.

By studying and understanding these examples, you can learn the techniques to use to effectively manage the space utilization, performance characteristics, and physical attributes of IMS DEDBs.

Example 1: Utilizing the SDEP part

The following figure shows example JCL statement for utilizing the SDEP part.

You can create a JCL stream for updating the SDEP History file and generating SDEP Utilization reports. A subset of the procedure can be used for report generation. This procedure should be run daily to update the History file.

Note: In update only mode (that is, the SYSIN DD statement is omitted, changed to DD DUMMY, or an empty data set is provided), a report is automatically generated if the threshold utilization value is exceeded.

The following figure shows example JCL.

```

//FABADA8 EXEC PGM=FABADA8,REGION=512K
//*****
//** REFORMAT SDEP DATA **
//*****
//SYSPRINT DD SYSOUT=A
//DADARI DD DSN=HPFP.SDEP.DATA,DISP=OLD
//DADARO DD DSN=HPFP.XSDEP.DATA,
//        DISP=(,CATLG,DELETE),
//        UNIT=SYSDA,
//        SPACE=(TRK,(5,2),RLSE),
//        DCB=BLKSIZE=4680
//*
//SORTSDEP EXEC PGM=SORT,COND=(4,LT)
//*****
//** SORT SDEP DATA **
//*****
//SYSOUT DD SYSOUT=A
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SORTIN DD DSN=HPFP.XSDEP.DATA,
//        DISP=(OLD,DELETE,KEEP)
//SORTOUT DD DSN=HPFP.XSDEP.SORTED.DATA,
//        DISP=(,CATLG,DELETE),
//        UNIT=SYSDA,
//        SPACE=(TRK,(5,2),RLSE)
//SYSIN DD *
//        SORT FIELDS=(1,24,CH,A)
//*
//*
//FABADA9 EXEC PGM=FABADA9,REGION=768K,COND=(4,LT)
//*****
//** UPDATE SDEP HISTORY AND GENERATE REPORTS **
//*****
//RPTOUT DD SYSOUT=A
//MSGOUT DD SYSOUT=A
//DADARI DD DSN=HPFP.XSDEP.SORTED.DATA,
//        DISP=(OLD,DELETE,KEEP)
//DASDHIO DD DSN=HPFP.SDEPHIST,DISP=OLD
//SYSIN DD *
//        DBDNAME=TSSDBD
//*
//

```

Figure 8. Example JCL for utilization process for SDEP part

The sample JCL updates the History file with all SDEP space utilization data journaled to date, and generates SDEP Utilization reports for all areas of database TSSDBD.

Example 2: Extracting SDEP space utilization data

These are example JCL statements for extracting SDEP space utilization data.

Two examples are presented for extracting SDEP space utilization data. The following figure is an example showing SDEP space utilization data that is to be extracted from area TSSAR01 of database TSSDBD.

```

//XTRACT EXEC FPUTIL,DBD=TSSDBD,
//        REST=00,RGN=256K
//*
//FPU.STEPLIB DD
//        DD DSN=HPFP.SHFPLMD0,DISP=SHR
//SCANCOPY DD DSN=HPFP.SDEP.DATA,
//        DISP=(MOD,KEEP,KEEP),
//        SPACE=(TRK,(5,2))
//SYSIN DD *
//        TYPE SCAN
//        ERROR SCAN
//        EXIT FABADA7
//        *
//        AREA TSSAR01
//        NOSORT
//        GO
//*

```

Figure 9. Extraction of SDEP space utilization data (First example)

The following figure is an example showing SDEP space utilization data that is to be extracted from areas TSSAR01 and TSSAR03 of database TSSDBD.

```
//XTRACT EXEC FPUTIL,DBD=TSSDBD,
//      REST=00,REGN=256K
//*
//FPU.STEPLIB DD
//      DD DSN=HPFP.SHFPLMD0,DISP=SHR
//SCANCOPY DD DSN=HPFP.SDEP.DATA,
//      DISP=(MOD,KEEP,KEEP),
//      SPACE=(TRK,(5,2))
//SYSIN DD *
TYPE SCAN
ERROR SCAN
*
AREA TSSAR01
EXIT FABADA7
NOSORT
GO
*
AREA TSSAR03
EXIT FABADA7
NOSORT
GO
/*
```

Figure 10. Extraction of SDEP space utilization data (Second example)

Related tasks

Running the SDEP Space Utilization utility process

To use the SDEP Space Utilization utility, you must run several programs. To run the programs, you must code JCL statements for those programs.

Example 3: Updating only the SDEP History file

The following figure shows example JCL statement for updating only the SDEP History file.

You can create a JCL stream for only to update the SDEP History file. You can modify the example JCL in [Figure 8 on page 21](#) so as to do the updating only by omitting the SYSIN DD statement in FABADA9 or changing it to:

```
//SYSIN DD DUMMY,DCB=BLKSIZE=80
```

This causes a warning message to be generated.

Example 4: Generating only SDEP Utilization reports

The following figure shows example JCL statement for generating only SDEP Utilization reports.

You can create a JCL stream for only to generate the SDEP Utilization reports. By running a subset of the SDEP update and report generation JCL ([Figure 8 on page 21](#)), it is possible to generate "SDEP Utilization" reports without running the following update phase:

1. Omit the DADARI DD statement in FABADA9 or change it to:

```
//DADARI DD DUMMY,DCB=BLKSIZE=72
```

2. Add the appropriate control statements.
3. Run FABADA9 only.

In the example shown in the following figure, SDEP Utilization reports are required for all areas of TSSDBD and areas TSS2AR01 and TSS2AR04 of database TSSDBD2.

```

//*****
//** GENERATE SDEP REPORTS ONLY **
//*****
//FABADA9 EXEC PGM=FABADA9,REGION=768K
//STEPLIB DD DSN=HPFP.SHFPLMD0,DISP=SHR
//MSGOUT DD SYSOUT=A
//RPTOUT DD SYSOUT=A
//DADARI DD DUMMY,DCB=BLKSIZE=72
//DASDHIO DD DSN=HPFP.SDEPHIST,DISP=OLD
//SYSIN DD *
DBDNAME=TSSDBD
DBDNAME=TSSDBD2,AREA=TSS2AR01
DBDNAME=TSSDBD2,AREA=TSS2AR04
/*

```

Figure 11. Example of generating only the SDEP Utilization reports

Chapter 3. Database Definition Record Create utility

Use the Database Definition Record Create utility (FABCUR5) to create the Database Definition Record (DURDBDFN), or to generate a report in the DURDBDFN file or a member in the IMS ACB library.

Topics:

- [“Functions of the Database Definition Record Create utility” on page 25](#)
- [“Data and system flow of the Database Definition Record Create utility” on page 25](#)
- [“Running the Database Definition Record Create utility” on page 25](#)
- [“DD statements for the Database Definition Record Create utility” on page 26](#)
- [“Input for the Database Definition Record Create utility” on page 28](#)
- [“Output for the Database Definition Record Create utility” on page 29](#)
- [“Example for the Database Definition Record Create utility” on page 33](#)

Functions of the Database Definition Record Create utility

The Database Definition Record Create utility (FABCUR5) provides a function to create the Database Definition Record (DURDBDFN) from the DBD-type DMB in the IMS ACB library, and to generate a report in the DURDBDFN file or to generate a DEDB DMB member in the IMS ACB library, or both.

The DURDBDFN data set is usually produced by the DEDB Unload Utility (FABCUR1) with the DEDB unloaded segment data set or by the DEDB Reload Segment Data Set Create utility (FABCUR6) with the DEDB reload segment data set. This Database Definition Record Create utility will be used when the DURDBDFN data set is lost.

Data and system flow of the Database Definition Record Create utility

This topic describes the data and system flow of the Database Definition Record Create utility.

The following figure shows the input to and output from FABCUR5.

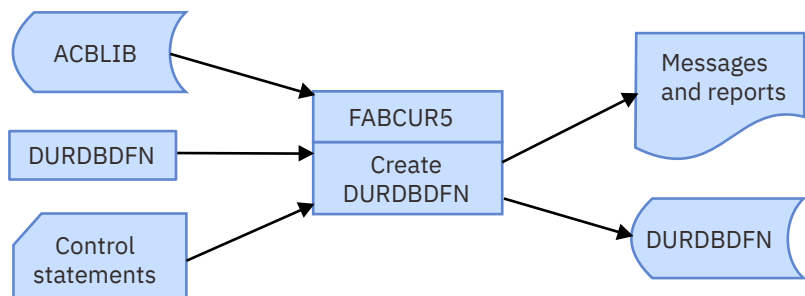


Figure 12. Flow of Database Definition Record Create utility

Running the Database Definition Record Create utility

The Database Definition Record Create utility (FABCUR5) is run as a standard z/OS batch job. An EXEC statement and DD statements that define the input and output data sets are required. FABCUR5 has the alias name FABEUR5.

Procedure

1. Code the JCL for the FABCUR5 job step.
2. Specify the DD statements to define input data sets, output data sets, and how the function is run.

3. Run the JCL.

Example

The following figure shows example JCL for FABCUR5.

```
//UR5      EXEC PGM=FABCUR5,REGION=rrrrM
//ACBLIB   DD DSN=IMSVS.ACBLIB,DISP=SHR
//DURDBDFN DD DSN=HPFP.UR.DURDBDFN,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(1,1))
//SYSPRINT DD SYSOUT=A
//REPORTS  DD SYSOUT=A
//SYSIN    DD *
... control statements ...
/*
```

Figure 13. Example JCL for FABCUR5

DD statements for the Database Definition Record Create utility

DD statements for the Database Definition Record Create utility (FABCUR5) determine the input and output data sets and how the utility is run.

The following table summarizes the DD statements of FABCUR5.

Table 7. FABCUR5 DD statements

DDNAME	Use	Format	Required or optional
ACBLIB	Input	PDS	Required (see note)
IMSACBA	Input	PDS	Optional
IMSACBB	Input	PDS	Optional
MODSTAT	Input		Optional
MODSTAT2	Input		Optional
DURDBDFN	Output or Input	Do not code DCB	Required (see note)
SYSPRINT	Output	LRECL=133	Required
REPORTS	Output	LRECL=133	Optional
SYSIN	Input	LRECL=80	Required

Note: When FUNCTION=PRINT is specified, this DD is optional.

All output data sets are blocked to the maximum size of the output device (unless overridden in the execution JCL). Since the blocking factor is determined at the execution time, standard labels must be used on all output data sets except SYSPRINT.

EXEC

The EXEC statement must be in the form:

```
// EXEC PGM=FABCUR5,REGION=rrrrM
```

But FABCUR5 has the alias name FABEUR5, therefore you can invoke FABCUR5 using the alias as follows:

```
// EXEC PRM=FABEUR5,REGION=rrrrM
```

rrrr

specifies the size of the region.

ACBLIB DD

Defines the library that contains the DMB for the database. This DD must be provided when FUNCTION=BUILD is specified. Otherwise this DD is optional.

IMSACBA DD

Defines the library that contains the DMB for the database. This DD must be provided if MODSTAT DD or MODSTAT2 DD is specified.

IMSACBB DD

Defines the library that contains the DMB for the database. This DD must be provided if MODSTAT DD or MODSTAT2 DD is specified.

MODSTAT DD

Defines the MODSTAT data set. When this DD is specified, the IMSACBA DD and IMSACBB DD statements must be specified instead of the ACBLIB DD.

MODSTAT2 DD

Defines the MODSTAT data set. When this DD is specified, the IMSACBA DD and IMSACBB DD statements must be specified instead of the ACBLIB DD.

DURDBDFN DD

Defines an output data set for the database definition record generated by FABCUR5 when FUNCTION=BUILD is specified. This contains the data extracted from the DMB that is used by the reload processor. The data set must reside on a direct-access device. Space requirements depend on the size of the DMB, but a couple of tracks usually suffice.

Do not code DCB information in your JCL. Do not specify DISP=MOD for this DD statement.

When FUNCTION=PRINT is specified, this statement defines the input data set that contains a formatted copy of the DMB, and it is optional.

SYSPRINT DD

Defines the output data set that contains messages issued by FABCUR5. The data set can reside on either a direct-access device or a printer; or it can be routed through the output stream. Do not code the DCB= parameter; it is recommended that you use:

```
//SYSPRINT DD SYSOUT=A
```

If REPORTS DD statement is not specified, the DBD Definition report is written in this data set.

REPORTS DD

Defines the output data set that contains the DBD Definition report. The data set can reside on either a direct-access device or a printer; or it can be routed through the output stream. Do not code the DCB= parameter; it is recommended that you use:

```
//REPORTS DD SYSOUT=A
```

If this DD statement is not specified, the DBD Definition report is written in the SYSPRINT DD data set.

SYSIN DD

This statement defines the input data set for the control statement. The data set can reside on a direct-access device or can be routed through the input stream.

Input for the Database Definition Record Create utility

You must specify the necessary input DD data sets to run FABCUR5.

SYSIN DD data set

The SYSIN data set contains the control cards that describe the processing to be done by the FABCUR5 utility.

Format

This control statement data set is usually the input stream. However, it can also be defined as a sequential data set or as a member of a partitioned data set. It must contain 80-byte, fixed-length records. Block size if coded, must be a multiple of 80.

This data set can contain several different types of control statements, including a comment statement. It can be coded as shown in the following figure.

```
//SYSIN DD *  
DBDNAME=DEDBJN22  
FUNCTION=BUILD  
/*
```

Figure 14. FABCUR5 SYSIN DD data set

Control statements

The FABCUR5 keywords and their associated parameter values can be coded in free format (columns 1 - 71). The syntactical rules are as follows:

1. Control statements are coded on 80-byte records.
2. Specifications for all control statements must start in column 1. A control statement record can include only one control statement.
3. A "keyword=value" specification must not span the control statement.
4. There must be one DBDNAME control statement, and it must be the first in the control statement stream.

DBDNAME control statement

The DBDNAME statement specifies the DBD name of the DEDB being reloaded. There must be only one DBDNAME statement. It contains the following keyword:

```
DBDNAME= dbdname
```

DBDNAME=

This statement specifies the DEDB DBD name for creating the Database Definition Record (DURDBDFN). It is used to access the ACB library or to check the DEDB definition record data set (DURDBDFN) when FUNCTION=PRINT is specified.

DBDNAME is a required keyword.

dbdname

Specifies the name of the DBD to be used.

FUNCTION control statement

The optional FUNCTION statement specifies that the FABCUR5 performs the Database Definition Record create function. It contains the following keywords:

```
[FUNCTION=BUILD|PRINT]
```

FUNCTION=

This optional statement specifies a function that the FABCUR5 performs.

BUILD

The Database Definition Record Create function has been performed. The DBD Definition report from the DMB member of the ACBLIB is created, too. This is the default value.

PRINT

The DBD Definition report has been created. Either ACBLIB DD or DURDBDFN DD, or both, must be provided.

Output for the Database Definition Record Create utility

The following topics describe the output from FABCUR5.

SYSPRINT DD data set

The SYSPRINT data set contains the messages issued by FABCUR5. The data set contains 133-byte records.

Format

If you code the block size in your JCL, it must be a multiple of 133. It is better to code your DD statement as follows:

```
//SYSPRINT DD SYSOUT=A
```

FABCUR5-Messages report

The following figure shows an example of the Messages report.

IMS HPFP UTILITIES - DEDBUR 5698-FPP	"FABCUR5 - MESSAGES" DATE: 11/22/2020 TIME: 11.43.21	PAGE: 1 FABCUR5 - V2R1
FABC0520I - CARD 1:DBDNAME=DEDBJN22		
FABC0501I - DATABASE DEFINITION RECORD FOR DBD: DEDBJN22 IS BUILT		
FABC0500I - FABCUR5 ENDED NORMALLY		

Figure 15. FABCUR5-Messages

REPORTS DD data set

The REPORTS data set contains the DBD Definition report. The data set contains 133-byte records.

Format

Do not code the DCB= parameter; it is recommended that you use:

```
//REPORTS DD SYSOUT=A
```

DBD Definition report

Purpose

The report provides the DBD definition information extracted from the DEDB definition record data set (DURDBDFN), or the DEDB DMB member in the ACBLIB library, or both.

Report content

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

ACBLIB DATA SET NAME: IMS.USER.ACBLIB

DBD NAME: DEDBJN22

RANDOMIZER: RMOD5

RECORD ORIGIN: --- (OLD:OLDACB NEW:NEWACB UTL:FABCUR5/FABCUR6 ---:ACBLIB)

IMS LEVEL: VERSION 14 RELEASE 1

DB LARGEST INFO:	CI-SIZE	UOW-1	NO(RAP'S/UOW)	UOW-2	SEG-LEN
	2,048	32,767	22,767	10,000	350

AREA:

AREA NO.	AREA NAME	CI-SIZE	UOW=	ROOT=	SDEP START		SDEP CI'S
					BLOCK#	RBA(HEX)	
1	DB22AR0	1,024	(5,1)	(5,1)	-	-	-
2	DB22AR1	512	(32767,10000)	(32767,10000)	-	-	-
3	DB22AR2	1,024	(13,3)	(15,3)	-	-	-
4	DB22AR3	1,024	(10000,1)	(10000,1000)	-	-	-
5	DB22AR4	2,048	(10,2)	(10,2)	-	-	-

SEGMENT:

```

+-----+
| LEGEND FOR SEGMENT INFORMATION |
+-----+
| R: ROOT SEGMENT  D: DDEP SEGMENT  S: SDEP SEGMENT |
| F: FIXED LENGTH  V: VARIABLE LENGTH I: COMP INIT   Y: PCL DEFINED |
+-----+

```

SEG. CODE	SEG. NAME	HIER LVL	PARENT S.CODE	TYPE	FIX VAR	PARENT		LENGTH		KEY		COMP-RTN	
						PCL	SSP	MAX	MIN	OFF	LEN	NAME	INIT
1	ROOTSEG1	1	-	R	F	-	-	300	300	2	6	DFSCMPX0	-
2	SDSEGNM1	2	1	S	V	-	-	350	20	-	-	-	-
3	DD1	2	1	D	V	-	-	320	20	2	7	DFSCMPX0	I

*** END OF DATABASE DEFINITION REPORT ***

Figure 16. DBD Definition report (Part 1 of 2)

DURDBDFN DATA SET NAME: HPFP.DEDBJN22.DURDBDFN

DBD NAME: DEDBJN22

RANDOMIZER: RMOD5

RECORD ORIGIN: UTL (OLD:OLDACB NEW:NEWACB UTL:FABCUR5/FABCUR6 ---:ACBLIB) FPS V2

IMS LEVEL: VERSION 14 RELEASE 1

DB LARGEST INFO:	CI-SIZE	UOW-1	NO(RAP'S/UOW)	UOW-2	SEG-LEN
	2,048	32,767	22,767	10,000	350

AREA:

AREA NO.	AREA NAME	CI-SIZE	UOW=	ROOT=	SDEP START		SDEP CI'S
					BLOCK#	RBA(HEX)	
1	DB22AR0	1,024	(5,1)	(5,1)	-	-	-
2	DB22AR1	512	(32767,10000)	(32767,10000)	-	-	-
3	DB22AR2	1,024	(13,3)	(15,3)	-	-	-
4	DB22AR3	1,024	(10000,1)	(10000,1000)	-	-	-
5	DB22AR4	2,048	(10,2)	(10,2)	-	-	-

SEGMENT:

```

+-----+
| LEGEND FOR SEGMENT INFORMATION |
+-----+
| R: ROOT SEGMENT  D: DDEP SEGMENT  S: SDEP SEGMENT |
| F: FIXED LENGTH  V: VARIABLE LENGTH I: COMP INIT   Y: PCL DEFINED |
+-----+

```

SEG. CODE	SEG. NAME	HIER LVL	PARENT S.CODE	TYPE	FIX VAR	PARENT		LENGTH		KEY		COMP-RTN	
						PCL	SSP	MAX	MIN	OFF	LEN	NAME	INIT
1	ROOTSEG1	1	-	R	F	-	-	300	300	2	6	DFSCMPX0	-
2	SDSEGNM1	2	1	S	V	-	-	350	20	-	-	-	-
3	DD1	2	1	D	V	-	-	320	20	2	7	DFSCMPX0	I

*** END OF DATABASE DEFINITION REPORT ***

Figure 17. DBD Definition report (Part 2 of 2)

Note: FABCUR5 adds "(XCI)" after the randomizer name in this report when an XCI randomizer is used.

ACBLIB DATA SET NAME**DURDBDFN DATA SET NAME**

The name of the data set from which the DBD definition information was extracted.

DBD NAME

The name of the data set from which the DBD definition information was extracted.

RANDOMIZER

The name of the randomizing module.

RECORD ORIGIN

The origin of the DBD definition information, as follows:

DBD NAME

The name of the DBD to be reported.

RANDOMIZER

The name of the randomizing module.

RECORD ORIGIN

the origin of the DBD definition information, as follows:

OLD

DURDBDFN was produced by FABCUR1 using OLD ACBLIB.

NEW

DURDBDFN was produced by FABCUR1 using NEW ACBLIB.

UTL

DURDBDFN was produced by FABCUR5 or FABCUR6.

The DBD definition information was extracted from the ACBLIB during FABCUR5 run.

One of following DURDBDFN level information is shown:

blank

DURDBDFN was produced by FABCUR5 or printed by using ACBLIB.

FPS Vn

DURDBDFN was produced by FPB of IMS HP Fast Path Utilities in IMS Fast Path Solution Pack Vn.

HFP Vn

DURDBDFN was produced by FPB of IMS HP Fast Path Utilities version n.

FPB V1

DURDBDFN was produced by FPB 1.x.

DBT V2

DURDBDFN was produced by IMS DBT 2.x.

IMS LEVEL

The IMS level of the source data set from which the DBD definition information was reported.

DB LARGEST INFO

The largest value of the areas in the DBD as follows:

CI-SIZE

The largest CI size of areas in the DBD.

UOW-1

The largest number of CIs per UOW of areas in the DBD.

NO(RAP'S/UOW)

The largest number of RAP CIs per UOW of areas in the DBD.

UOW-2

The largest number of DOVF CIs per UOW of areas in the DBD.

SEG-LEN

The largest length of segments of areas in the DBD.

AREA NO.

The area number.

AREA NAME

The area name.

CI-SIZE

The CI size of the area.

UOW=

The UOW= parameter value for the area defined in the DBD.

ROOT=

The ROOT= parameter value for the area defined in the DBD.

SDEP START BLOCK#

The first block number of CI in the SDEP part of the area.

SDEP START RBA (HEX)

The hexadecimal value of the first CI in the SDEP part of the area.

SDEP CI'S

The number of SDEP CIs in the area data set. This value is reported only when DBD definition information was extracted from DURDBDFN created by OLD ACBLIB.

SEG.CODE

The segment code.

SEG.NAME

The segment name.

HIER LVL

The hierarchical level of the segment.

PARENT S.CODE

The parent segment code of the segment.

TYPE

The type of the segment as follows:

R

The root segment.

S

The sequential dependent segment.

D

The direct dependent segment.

FIX|VAR

The attribute of the segment as follows:

F

The fixed-length segment.

V

The variable-length segments.

PARENT PCL

Determines whether the parent of the segment has a PCL pointer.

Y means that the parent of the segment has a PCL pointer.

PARENT SSP

Determines the number of subset pointers that the parent of the segment has, if any is defined.

LENGTH MAX

The maximum length of the segment defined in the DBD.

LENGTH MIN

The minimum length of the segment defined in the DBD.

KEY OFF

The offset of the key field of the segment, if any is defined.

KEY LEN

The key length of the segment, if any is defined.

COMP-RTN NAME

The name of the segment edit/compression routine of the segment, if any is defined.

COMP-RTN INIT

Determines whether COMPRTN= INIT subparameter has been defined in the DBD.

I means that the COMPRTN= INIT subparameter has been defined.

Example for the Database Definition Record Create utility

The following figure shows example JCL statement for the Database Definition Record Create utility.

The example shown in the following figure selects the current active ACB library, which is used in IMS XRF or in the online change environment. Instead of an ACBLIB DD statement, IMSACBA DD and IMSACBB DD statements are used. MODSTAT DD and MODSTAT2 DD (in the XRF environment) statements are specified.

```
//UR5      EXEC PGM=FABCUR5,REGION=iiiiM
//IMSACBA  DD DSN=IMSVS.ACBLIBA,DISP=SHR
//IMSACBB  DD DSN=IMSVS.ACBLIBB,DISP=SHR
//MODSTAT  DD DSN=IMSVS.MODSTAT,DISP=SHR
//MODSTAT2 DD DSN=IMSVS.MODSTAT2,DISP=SHR
//DURDBDFN DD DSN=HPFP.UR.DURDBDFN,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(1,1))
//SYSPRINT DD SYSOUT=A
//REPORTS  DD SYSOUT=A
//SYSIN DD *
... control statements ...
/*
```

Figure 18. Sample of FABCUR5 DD statements selects the current ACB libraries

Chapter 4. DEDB Reload Segment Data Set Create utility

Use the DEDB Reload Segment Data Set Create utility (FABCUR6) to format and write the segment data records in the format required by the FPA Reload function and the FPB DEDB Reload utility.

Topics:

- [“Functions of the DEDB Reload Segment Data Set Create utility” on page 35](#)
- [“Data and system flow of the DEDB Reload Segment Data Set Create utility” on page 35](#)
- [“Calling the DEDB Reload Segment Data Set Create utility \(from your program\)” on page 36](#)
- [“DD statements for the DEDB Reload Segment Data Set Create utility” on page 39](#)
- [“Input for the DEDB Reload Segment Data Set Create utility” on page 42](#)
- [“Output for the DEDB Reload Segment Data Set Create utility” on page 46](#)
- [“Setting site default values for the DEDB Reload Segment Data Set Create utility” on page 48](#)
- [“Examples for the DEDB Reload Segment Data Set Create utility” on page 50](#)

Functions of the DEDB Reload Segment Data Set Create utility

The DEDB Reload Segment Data Set Create utility (FABCUR6) is called from a user application program. FABCUR6 is used to format and write the segment data records in the format that is required by the FPA Reload function and the FPB DEDB Reload Utility (FABCUR3).

The user application program does not need to handle:

- Invoking the randomizing program for each root segment
- Formatting the segment record
- Writing the segment records to an output data set

FABCUR6 is link-edited into a user program, or can be invoked dynamically using ATTACH, LINK, or DYNAMIC CALLs. The IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0) must be concatenated to your application program library in the JOBLIB/STEPLIB DD statement.

Data and system flow of the DEDB Reload Segment Data Set Create utility

This topic describes the data and system flow of the DEDB Reload Segment Data Set Create utility.

The following figure shows the flow of FABCUR6.

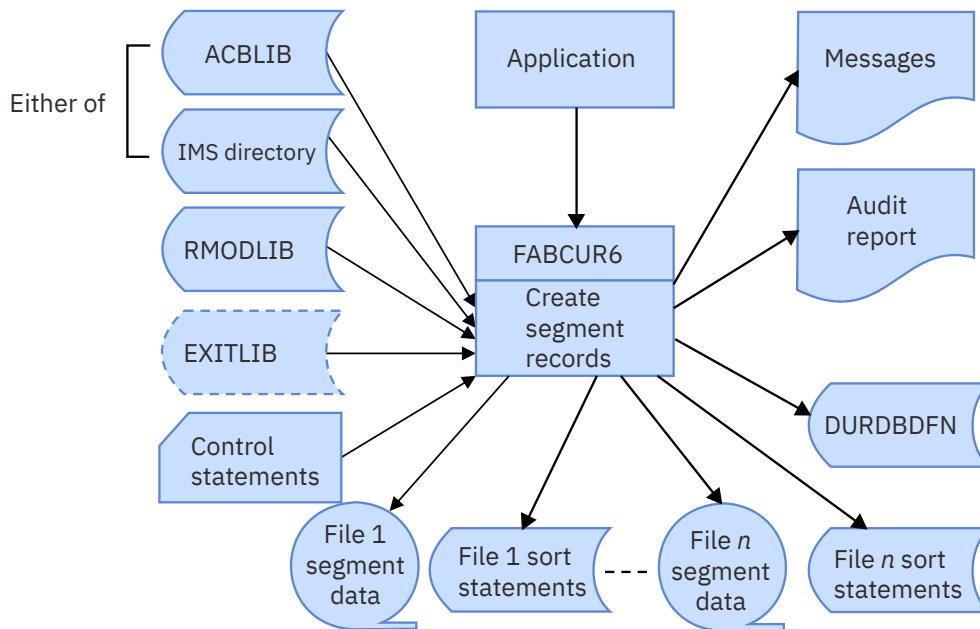


Figure 19. Flow of DEDB Reload Segment Data Set Create utility

Calling the DEDB Reload Segment Data Set Create utility (from your program)

FABCUR6 can be called from programs written in any language conforming to the z/OS register linkage conventions.

Example

When calling, code the JCL as shown in the following figure.

```

//UR6      EXEC PGM=appl-name,REGION=rrrrM
//          DD dd statements necessary for an application
//ACBLIB    DD DSN=IMSVS.ACBLIB,DISP=SHR
//RMODLIB   DD DSN=IMSVS.PGMLIB,DISP=SHR
//EXITLIB   DD DSN=USER.PGMLIB,DISP=SHR
//UR6DBDFN DD DSN=HPFP.UR.DURDBDFN,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,SPACE=(TRK,(1,1))
//DURDzzz0 DD DSN=HPFP.UR.FILEzzz.SEGDATA,
//          DISP=(NEW,CATLG,DELETE),UNIT=TAPE
//DURDzzzE DD DSN=HPFP.UR.FILEzzz.SEGDATAE,
//          DISP=(NEW,CATLG,DELETE),UNIT=TAPE
//DURSzzz0 DD DSN=HPFP.UR.FILEzzz.SORTCARD,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,SPACE=(TRK,(1,1))
//UR6PRINT DD SYSOUT=A
//UR6AUDIT  DD SYSOUT=A
//UR6CTL    DD *
... control statements ...
/*

```

Figure 20. Sample JCL for FABCUR6

Dynamic linkage to an application

FABCUR6 can be invoked dynamically via ATTACH, LINK, or DYNAMIC CALLs.

An application can also use the alias name FABEUR6. The IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0) must be concatenated to your application program library in the JOBLIB/STEPLIB DD statement.

Static linkage to an application

FABCUR6 can be invoked statically.

FABCUR6 must be included from the IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0) when you link edit your program that calls FABCUR6. Another entry point name, FABEUR6, can be used by your program to call FABCUR6.

Application interface

FABCUR6 is called with a parameter list containing a function code and one or more other data items depending on the function code.

The function codes are as shown in the following table:

Function code	Description
INIT	Initialization
PUT	Build and write segment
EOF	End of file

Initialization function: INIT

This topic describes the initialization function.

INIT performs the following functions:

- Sets up the required environment
- Reads necessary DMB member from the ACB library and builds the DEDB Data Base Definition control block for a DURDBDFN record
- Edits and parses the control statements
- Opens all required output data sets
- Loads the user exit routine, if necessary
- Load the randomizing routine. If it is the XCI randomizer, run the XCI Initialization call to the randomizer.
- Loads and performs a DEDB area open call (entry code 12) to the segment edit/compression routines, if necessary

The DEDB Reload Segment Data Set Create utility (FABCUR6) allows you to specify site default parameters. Macros and sample JCL streams are provided to generate the site default table.

Note: The INIT function must be performed prior to any other function call.

INIT parameter list

Function Code

This is a mandatory parameter. It is a 4-byte field containing an address of a 4-byte function code field containing 'INIT'.

DBD Name

This is a mandatory parameter. It is a 4-byte field containing the address of an 8-byte character field that contains the DBD name of the DEDB being processed (that is, the member name of ACBLIB).

Coding INIT in a COBOL program

In WORKING_STORAGE, define the following:

```
01 FUNC_INIT    PIC X(04)    VALUE 'INIT' .
01 DBDNAME      PIC X(08)    VALUE 'dbdname' .
```

where *dbdname* is the DBD name.

In the PROCEDURE DIVISION, add the following:

```
CALL 'FABCUR6' USING FUNC_INIT,
                      DBDNAME.
```

Build and write segment: PUT

This topic describes the build and write segment.

PUT performs the following functions:

- Invokes the randomizing routine if the segment being written is a root segment
- Builds a segment prefix
- Performs a CMP (compress) function call to an exit routine if the EXITRTN control statement is provided
- Performs a segment compression call (entry code 0) if the IMSCOMP=CMP is provided and the COMPRTN parameter is defined in the DBD to the segment
- Writes the segment record to the appropriate output data set

Note: Segments within a database record must be provided to FABCUR6 in hierarchical order.

PUT parameter list

Function code

This is a mandatory parameter. It is a 4-byte field containing an address of a 4-byte function code field containing PUT, left-adjusted in the field.

Segment I/O area

This is a mandatory parameter. It is a 4-byte field containing an address of an I/O area of a segment data being written.

The following table shows the layout of the Segment I/O Area.

Table 8. Layout of segment I/O area

Field name	Assembly code definition	Description
PSEG_NAME	CL8	Segment name
PSEG_SSPTRS	8CL1	Subset pointer indicator
PSEG_DATA	0XLnnn	Segment data including the 2-byte length field
PSEG_DATA_LL	XL2	Segment data length

Note: The subset pointer indicator is eight 1-byte positional flags set left to right for pointers 1 - 8 that specify whether or not this segment is the target of a subset pointer. Setting 'Y' in one of the flag bytes will cause the appropriate subset pointer to be set.

Coding PUT in a COBOL program

In WORKING_STORAGE, define the following:

```
01  FUNC_PUT                PIC X(04)  VALUE 'PUT '.
01  IO_AREA.
    05  SEG_NAME             PIC X(08) .
    05  SEG_SSPTRS           PIC X      OCCURS 8 TIMES.
    05  SEG_DATA             PIC X(nnn) .
    05  PSEG_DATA            REDEFINES SEG_DATA.
    07  PSEG_DATA_LL         PIC S9(4)  COMP.
    07  PSEG_DATA_CONTAIN    PIC X(nnn-2) .
```

where *nnn* is the length of the longest segment defined in DBDGEN.

In the PROCEDURE DIVISION, add the following:

```
CALL  'FABCUR6' USING FUNC_PUT,
                        IO_AREA.
```

End of file: EOF

This function is invoked after all segments for a database have been processed. EOF performs following functions:

- Generates the audit report
- Generates the sort control statements
- If it is the XCI randomizer, run the XCI Termination call to the randomizer.
- Performs a DEDB area close call (entry code 16) to segment edit/compression routines, if necessary
- Performs an END (clean up) function call to an exit routine, if the EXITRTN control statement is provided
- Closes all output data sets

EOF parameter list

Function code

This is a mandatory parameter. It is a 4-byte field containing an address of a 4-byte function code field containing EOF, left-adjusted in the field.

Coding EOF in a COBOL program

In WORKING_STORAGE, define the following:

```
01 FUNC_EOF PIC X(04) VALUE 'EOF'.
```

In the PROCEDURE DIVISION, add the following:

```
CALL  'FABCUR6' USING FUNC_EOF.
```

DD statements for the DEDB Reload Segment Data Set Create utility

DD statements for the DEDB Reload Segment Data Set Create utility (FABCUR6) determine the input and output data sets and how the utility is run.

The following table shows the JCL DD statements required to be included in the job step.

Table 9. FABCUR6 DD statements

DDNAME	Use	Format	Required or optional
JOBLIB/STEPLIB	Input	PDS	Required
ACBLIB	Input	PDS	Optional
IMSACBA	Input	PDS	Optional
IMSACBB	Input	PDS	Optional
MODSTAT	Input		Optional
MODSTAT2	Input		Optional
RMODLIB	Input	PDS	Required
EXITLIB	Input	PDS	Optional
UR6CTL	Input	LRECL=80	Optional
UR6PRINT	Output	LRECL=133	Required

Table 9. FABCUR6 DD statements (continued)

DDNAME	Use	Format	Required or optional
UR6AUDIT	Output	LRECL=133	Required
UR6DBDFN	Output	Do not code DCB	Optional
DURDzzzO or XDzzzzzO	Output	Do not code DCB except BLKSIZE allowed	Optional
DURDzzzE or XDzzzzzE	Output	Do not code DCB except BLKSIZE allowed	Optional
DURSzzzO or XSzzzzzO	Output	LRECL=80	Optional

All output data sets are blocked to the maximum size of the output device (unless overridden in the execution JCL). Since the blocking factor is determined at execution time, standard labels must be used on all output data sets except UR6PRINT and UR6AUDIT.

JOBLIB/STEPLIB DD

Defines the library that contains the user application program.

The following libraries must be concatenated to your application program library in the JOBLIB/STEPLIB DD statement:

- The IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0).
- The IMS Tools Base library (SGLXLOAD), if you specify the IMSCATHLQ=*bsdshlq* keyword.

ACBLIB DD

Defines the library that contains the DMB for the database. This DD must be provided when IMSCATHLQ=*NO.

If MODSTAT/MODSTAT2 DD is provided, this DD is not necessary.

If you specify the IMSCATHLQ=*bsdshlq* keyword, ACBLIB DD statement is not used. The IMS directory is used instead of the ACB library.

IMSACBA DD

Defines the library that contains the DMB for the database. This DD must be provided if MODSTAT/MODSTAT2 DD is specified.

IMSACBB DD

Defines the library that contains the DMB for the database. This DD must be provided if MODSTAT/MODSTAT2 DD is specified.

MODSTAT DD

Defines the MODSTAT data set. When this DD is specified, the IMSACBA and IMSACBB DD must be specified instead of the ACBLIB DD.

MODSTAT2 DD

Defines the MODSTAT2 data set. When this DD is specified, the IMSACBA and IMSACBB DD must be specified instead of the ACBLIB DD.

RMODLIB DD

Defines the library that contains the randomizing routine and/or segment edit/compression routine.

Instead of defining the library on this DD statement, the library can be concatenated on the STEPLIB/JOBLIB DD statement.

EXITLIB DD

Defines the library that contains the program load module specified in the EXITRTN control statement.

UR6CTL DD

Defines the control statement input data set. This data set can reside on a direct-access device, or be routed through the input stream.

UR6PRINT DD

Defines the output data set that contains messages issued by FABCUR6. The data set can reside on a direct-access device or printer, or be routed through the output stream. You can code RECFM=FBA, LRECL=133 on your DD statement, but it is better to use:

```
//UR6PRINT DD SYSOUT=A
```

UR6AUDIT DD

Defines the output data set that contains the FABCUR6 Audit Control report. The data set can reside on a direct-access device or printer, or be routed through the output stream. You can code RECFM=FBA, LRECL=133 on your DD statement, but it is better to use:

```
//UR6AUDIT DD SYSOUT=A
```

UR6DBDFN DD

Defines an output data set for the database definition record generated by FABCUR6. This contains the data extracted from the DMB that is used by the reload processor. The data set must reside on a direct-access device. Space requirements depend on the size of the DMB, but a two tracks are usually enough. Do not code DCB information in your JCL.

Do not specify DISP=MOD for this DD statement.

If OUTDD=NO is specified in UR6CTL DD, this DD is optional.

DURDzzzO or XDzzzzzO DD

Defines an output data set for all of the database segment records produced for one or more of the areas defined in the DMB. A DURDzzzO DD statement is for areas in the range of 1 - 999, and an XDzzzzzO DD statement is for areas in the range of 1 - 2048. If the area number of the unloaded area is greater than 999, you should provide the XDzzzzzO DD statement. The value of zzz or zzzzz is made up of right-aligned digits, with leading zeros if needed.

DCB attributes are calculated by FABCUR6. RECFM is VB, and the default block size is the maximum block size of the output device. A block size override may be specified on the DD statement. Do not code any other DCB parameters in your JCL.

Do not specify DISP=MOD for these DD statements.

If OUTDD=NO is specified in UR6CTL DD, this DD is optional.

The rules for supplying the DURDzzzO or the XDzzzzzO data sets are discussed in the topic "FILECTL control statement" (for DEDB Unload) in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

DURDzzzE or XDzzzzzE DD

Defines the second copy data set for DURDzzzO or XDzzzzzO. For a DURDzzzE or XDzzzzzE DD statement, there must be a corresponding DURDzzzO or XDzzzzzO DD statement. A DURDzzzE DD statement is for areas in the range of 1 - 999, and an XDzzzzzE DD statement is for areas in the range of 1 - 2048. If the area number of the unloaded area is greater than 999, you should provide the XDzzzzzE DD statement. The value of zzz or zzzzz is made up of right-aligned digits, with leading zeros if needed.

DCB attributes are calculated by FABCUR6. RECFM is VB, and the default block size is the maximum block size of the output device. A block size override may be specified on the DD statement. Do not code any other DCB parameters in your JCL.

Do not specify DISP=MOD for these DD statements.

DURSzzzO or XSzzzzzO DD

Defines an output data set that contains the SORT control statements for the segment data set associated with it. There must be a DURSzzzO or XSzzzzzO data set for each DURDzzzO or XDzzzzzO data set. The data set must reside on a direct-access device. This data set is required even if the corresponding DURDzzzO or XDzzzzzO data set does not need to be sorted. A DURSzzzO DD statement is for areas in the range of 1 - 999, and an XSzzzzzO DD statement is for areas in the range of 1 - 2048. If the area number of the unloaded area is greater than 999, you should provide the XSzzzzzO

DD statement. The value of zzz or zzzzz is made up of right-aligned digits, with leading zeros if needed. The space required is very small; one track suffices.

The DCB information is hard-coded in FABCUR6. Do not code the DCB information in your JCL.

Do not specify DISP=MOD for these DD statements.

If OUTDD=NO is specified in UR6CTL DD, this DD is optional.

Input for the DEDB Reload Segment Data Set Create utility

You must specify DD statements for the job control language (JCL) to run the DEDB Reload Segment Data Set Create utility.

UR6CTL DD data set

The UR6CTL data set contains the user's description specifying FABCUR6 to create the reload segment records.

Note: The default values for FABCUR6 control statement can be changed by using the site default table. For more information, see the topic "Site default support for FPB" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

Format

This control statement data set usually resides in the input stream. However, it can also be defined as a sequential data set or as a member of a partitioned data set. It must contain 80-byte, fixed-length records. Block size, if coded, must be a multiple of 80. This data set can contain several different types of control statements, including a comment statement. It can be coded as shown in the following figure.

```
//UR6CTL DD *  
FILECTL=1,ALL  
EXITRTN=exit-routine  
/*
```

Figure 21. FABCUR6 UR6CTL data set

Control statements

The FABCUR6 control statements are:

- FILECTL
- EXITRTN
- IMSCOMP
- USERCTL
- FORMAT
- AREA_INFORMATION_RECORD
- LRECL
- OUTDD
- IMSCATHLQ
- IMSCATACB_INPUT

The control statements are read in from the input source specified in the UR6CTL DD statement.

For information about the syntax of these control statements, see the topic "DEDB Unload SYSIN DD data set control statements" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

FILECTL control statement

The optional FILECTL statement controls grouping of multiple areas' segment data into a single output file.

```
[FILECTL={zzzzz},{ALL|x|(x,y,...)|(x-y)|(*)}]
```

FILECTL=

Here zzzzz is a decimal number 1 - 2048. This statement is composed of the following:

zzzzz

Specifies the output file number described by this control statement. The number is specified as a 1-to-5 digit decimal number. There must be related DD statements that have this number for an unloaded file, DURDzzzO and an output data set for the SORT control statement, DURSzzzO in the JCL stream for each data set specified on a FILECTL control statement. The value of zzz or zzzzz in the DD name is made up of right-aligned digits, with leading zeros if need.

ALL

Writes the segment data records for all output areas into the DURDzzzO data set.

x

Writes the segment data records for output area x into the DURDzzzO data set.

(x,y,...)

Writes the segment data records for output area x, area y, ... into the DURDzzzO data set.

(x-y)

Writes the segment data records for output area x, area x+1, ..., area y into the DURDzzzO data set.

(*)

Writes the segment data records for all output areas that are not specified on other FILECTL= control statements into the DURDzzzO data set.

Area numbers x and y are decimal numbers in the range of 1 - 2048.

Instead of DURDzzzO, you can specify XDzzzzzO.

Instead of DURDzzzE, you can specify XDzzzzzE if it is specified.

Instead of DURSzzzO, you can specify XSzzzzzO.

Empty area considerations

When there are no segment data records for output areas specified in FILECTL statement, the areas are regarded as empty. They are initialized during the reload, unless EMPTY=NO is specified on the USERCTL control statement. This consideration is applicable when the area is specified in FILECTL as in the following:

- 'ALL' is selected in the FILECTL statement.
- '*' is selected in the FILECTL statement.
- All areas are specified clearly in the FILECTL statement.

Default values

If FILECTL statements are not specified, each output area (number zzzzz) corresponds to its own DURDzzzO or XDzzzzzO data set. The DURDzzzO or XDzzzzzO data set contains all segments that FABCUR3 loads into area zzzzz, where zzzzz is the area number (field DMACRAID in the DMAC control block) assigned to that area during the processing of ACBGEN.

Error conditions

Duplicate references to an area or file in the FILECTL control statements are flagged with an error message, and cause program termination.

EXITRTN control statement

The optional EXITRTN statement specifies the name of the user exit routine that will be invoked with a COMPRESS function.

```
[EXITRTN=exit-routine]
```

EXITRTN=

This optional keyword specifies the name of the user exit routine to be invoked with a COMPRESS function.

exit-routine

Identifies the name of the user exit routine that will be called.

An EXITLIB DD statement must be provided when this control statement is specified.

For more information, see the topic "Exit routine option and its interface" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

IMSCOMP control statement

The optional IMSCOMP statement specifies whether the segment edit/compression routine will be invoked with segment compression call (entry code 0) for candidate Segments.

```
[IMSCOMP={NO|CMP}]
```

IMSCOMP=

This optional keyword determines whether the segment edit/compression routine will be invoked with segment compression call (entry code 0) for candidate segment.

NO

Specifies that the segment edit/compression routine is not invoked. IMSCOMP=NO is the default value.

CMP

Specifies that the segment edit/compression routine is invoked with segment compression call (entry code 0) for candidate segments.

USERCTL control statement

The optional USERCTL statement controls SDEP segment sequencing and whether FABCUR3 is to initialize empty areas.

```
USERCTL  
[SDEPSEQ={NORMAL|USER}]  
[EMPTY={NO|YES}]
```

USERCTL

This control statement has the following keywords.

SDEPSEQ=

This optional keyword controls the SDEP sequencing.

NORMAL

Specifies that the first SDEP segment inserted by FABCUR6 will be the first SDEP segment retrieved by IMS using GN processing. SDEPSEQ=NORMAL is the default value.

USER

Specifies that the first SDEP segment inserted by FABCUR6 will be the last SDEP segment retrieved by IMS using GN processing.

EMPTY=

This optional keyword determines whether area information records will be generated for areas that have no segment that causes FABCUR3 to initialize empty areas.

NO

Specifies that no area information records will be generated for areas that have no segment, and FABCUR3 will not initialize empty areas. EMPTY=NO is the default value.

YES

Specifies that area information records will be generated for areas that have no segment, and FABCUR3 will initialize empty areas.

FORMAT control statement

The optional FORMAT statement specifies the type of reloaded segment record format.

```
[FORMAT={DBT | TFMT}]
```

FORMAT=

The control statement specifies the type of reloaded segment record format.

DBT

Specifies that the format of the reloaded segment records is the same as IMS DBT 2.x.
FORMAT=DBT is the default value.

TFMT

Specifies that the format of the reloaded segment records is enhanced, which means that the prefix part of the record is generated based on the maximum number of segment levels defined in DBD.

AREA_INFORMATION_RECORD control statement

The optional AREA_INFORMATION_RECORD statement specifies whether the area information record will be generated for areas.

```
AREA_INFORMATION_RECORD={YES | NO}
```

AREA_INFORMATION_RECORD=

The optional AREA_INFORMATION_RECORD statement specifies whether the area information record will be generated for areas.

YES

Specifies that area information records will be generated for areas. For empty areas, the EMPTY= keyword parameter of the USERCTL control statement decides whether to generate area information records for the areas. AREA_INFORMATION_RECORD=YES is the default value.

NO

Specifies that no area information records will be generated for areas. The EMPTY=YES keyword parameter of the USERCTL control statement will be ignored except when AREA_INFORMATION_RECORD=NO, FORMAT=TFMT, and LRECL=SEGTFMT control statements are specified.

The segment records that are created by FABCUR6 with the AREA_INFORMATION_RECORD=NO option can be reloaded with the FPB DEDB Reload utility, but such records cannot be reloaded with the FPA Reload function.

An abbreviation AIR is used for AREA_INFORMATION_RECORD.

LRECL control statement

```
LRECL={BLOCK | SEGTFMT}
```

LRECL=

The LRECL statement is composed of:

BLOCK

LRECL of a reload segment data set generated by FABCUR6 is determined as BLKSIZE - 4.
LRECL=BLOCK is the system default value.

SEGTFMT

LRECL is determined on the basis of the maximum length of segments and the maximum number of segment levels defined in DBD by specifying LRECL=SEGTFMT together with the FORMAT=TFMT control statement.

OUTDD control statement

The OUTDD statement specifies whether the DD statements for the output data sets are mandatory.

```
OUTDD={YES | NO}
```

OUTDD=

This optional statement specifies whether the DD statements for the output data sets are mandatory.

YES

Specifies that the DD statement for the output data sets, which is DURDzzzO, XDzzzzzO, DURSzzzO, XSzzzzzO, or UR6DBDFN, is required. OUTDD=YES is the default value.

NO

Specifies that the DD statement for the output data sets, which is DURDzzzO, XDzzzzzO, DURSzzzO, XSzzzzzO, or UR6DBDFN, is optional.

IMSCATHLQ control statement

The optional IMSCATHLQ statement specifies the high-level qualifier of the bootstrap data set of the IMS directory, which is an extension of the IMS catalog. You must enable the IMS catalog and the IMS management of ACBs when you specify the high-level qualifier of the bootstrap data set of the IMS directory.

```
IMSCATHLQ={*NO|bsdshlq}
```

IMSCATHLQ=

This control statement has the following keywords:

bsdshlq

Reads the ACB member from the IMS directory instead of the ACB library by using IMS Tools Catalog Interface. *bsdshlq* specifies the high-level qualifier of the IMS directory bootstrap data set.

***NO**

Reads the ACB member from the ACB library. IMSCATHLQ=*NO is the default value.

IMSCATACB_INPUT control statement

The optional IMSCATACB_INPUT statement specifies whether to retrieve the currently active ACB definition or the pending ACB definition from the IMS directory. This statement is effective only when the IMSCATHLQ=*bsdshlq* statement is specified.

```
IMSCATACB_INPUT={CURRENT|PENDING}
```

IMSCATACB_INPUT=

This control statement has the following keywords:

CURRENT

The currently active ACB member is retrieved from the IMS directory data sets. IMSCATACB_INPUT=CURRENT is the default value.

PENDING

The pending ACB member is retrieved from the staging data set.

Output for the DEDB Reload Segment Data Set Create utility

The following topics describe the output from FABCUR6.

UR6PRINT DD data set

The UR6PRINT data set contains the messages issued by the FABCUR6 program. This data set contains 133-byte records.

Format

If you code the block size in your JCL, it must be a multiple of 133. It is better to code your DD statement as follows:

```
//UR6PRINT DD SYSOUT=A
```

FABCUR6-Messages report

The following figure shows an example of the Messages report.

```
IMS HPFP UTILITIES - DEDBUR                "FABCUR6 - MESSAGES"                PAGE: 1
5698-FPP                                DATE: 11/22/2020 TIME: 20.45.03          FABCUR6 - V2R1

FABC0620I - CARD 1: FILECTL=1,1
FABC0603W - NO SEGMENTS WILL BE RELOADED TO AREA 2 (AREANAME: DB22AR1)
FABC0603W - NO SEGMENTS WILL BE RELOADED TO AREA 3 (AREANAME: DB22AR2)
FABC0603W - NO SEGMENTS WILL BE RELOADED TO AREA 4 (AREANAME: DB22AR3)
FABC0603W - NO SEGMENTS WILL BE RELOADED TO AREA 5 (AREANAME: DB22AR4)
FABC0600W - FABCUR6 ENDED WITH WARNINGS
```

Figure 22. FABCUR6-Messages report

UR6AUDIT DD data set

The DEDB Reload Segment Data Set Create utility (FABCUR6) generates a two-part Audit Control report to provide verification totals. This data set contains 133-byte records.

Format

If you code the block size in your JCL, it must be a multiple of 133. It is better to code your DD statement as follows:

```
//UR6AUDIT DD SYSOUT=A
```

FABCUR6 Audit Control report

The FABCUR6 Audit Control report has the following parts:

1. Part 1: SEGMENTS TO BE RELOADED TO DATABASE dbdname

This section of the report provides a count of the number of segments that are to be reloaded to each area of the new database and to the database total.

2. Part 2: SEGMENT TOTALS BY OUTPUT FILE

This section of the report provides segment counts and area totals by the output file ddname. File totals and a database total are also provided.

The area totals should match the area totals in Part 1. The file totals are ultimately verified against the reload file totals. The database total should match the total in Part 1.

If FILECTL statements are not used (that is, ddnames default to area numbers), there is only one area per file. Conversely, if FILECTL statements are used, a file may contain data for more than one area.

The following figure shows an example of the Audit Control report.

SEGMENTS TO BE RELOADED TO DATABASE DEDBJN22 :

AREA NO	AREA NAME	SEG CODE	SEG NAME	NUMBER OF SEGMENTS
1	DB22AR0	1	ROOTSEG1	2
		2	SDSEGNM1	0
		3	DD1	6
		4	DD2	6
		5	DD3	7
		6	DD4	4
		7	DD43	2
		8	DD44	0
		9	DD5	1
		10	DD53	0
** AREA TOTALS **				28
*** DATABASE TOTAL ***				28

Figure 23. FABCUR6 Audit Control report (Segments to be reloaded to database)

The following figure shows an example of the segment totals by output file section of the Audit Control report.

SEGMENT TOTALS BY OUTPUT FILE:

FILE DDNAME	AREA NO	SEG CODE	SEG NAME	NUMBER OF SEGMENTS
DURD0010	1	1	ROOTSEG1	2
		2	SDSEGNM1	0
		3	DD1	6
		4	DD2	6
		5	DD3	7
		6	DD4	4
		7	DD43	2
		8	DD44	0
		9	DD5	1
		10	DD53	0
** AREA TOTALS **				28
*** FILE TOTALS ***				28
**** DATABASE TOTAL ****				28

Figure 24. FABCUR6 Audit Control report (Segment totals by output file)

Setting site default values for the DEDB Reload Segment Data Set Create utility

The DEDB Reload Segment Data Set Create utility allows you to specify site default values. Macros and sample JCL streams are provided to generate the site default table.

If you want to change the default values for control statements, use macro FABCOP6M and sample JCL FABCOP6J and generate a site default table.

The generated site default table library must be concatenated to the IMS HP Fast Path Utilities load module library in the JOBLIB or STEPLIB DD statement.

Use the TABLESET= parameter to specify the type of the table to generate. The keywords for the TABLESET= parameter are as follows:

USER

Builds a site default table. This is the default value.

SYSTEM

Builds a system default table that is to be used internally by the FABCUR6 program. Users of FABCUR6 should not specify this value.

DSECT

Builds a DSECT to map default table entries. Users of FABCUR6 should not specify this value.

When coding the macros, note the following:

- Under TABLESET=USER, specifying system default value will cause FABD3675I message to be generated and a table entry for the keyword value will not be generated.
- Under TABLESET=USER, coding the same macro more than once will cause FABD3676E message to be generated and will end with return code of 8. All necessary site default values for a macro must be specified in the same macro.

FABCOP6M macro

The following control statements can be specified:

IMSCOMP= or DILCOMP=

Specifies whether the segments of the unloaded record should be compressed if the segment edit/compression routine is defined for the segment in DBDGEN.

CMP

Specifies that the unloaded record should contain compressed segments.

N | NO

Specifies that the unloaded record should contain segments that are expanded. This is the system default value.

SDEPSEQ=

Specifies the type of the sequencing for the SDEP segment.

NORMAL

Specifies that the first SDEP segment inserted by FABCUR6 will be the first SDEP segment retrieved by IMS using GN processing. This is the system default value.

USER

Specifies that the first SDEP segment inserted by FABCUR6 will be the last SDEP segment retrieved by IMS using GN processing.

EMPTY=

Specifies whether area information records will be generated for areas that have no segment that causes FABCUR3 to initialize empty areas.

Y | YES

Specifies that area information records will be generated for areas that have no segment, and FABCUR3 will initialize empty areas.

N | NO

Specifies that no area information records will be generated for areas that have no segment, and FABCUR3 will not initialize empty areas. This is the system default value.

FORMAT=

Specifies the format of the unloaded segment records.

DBT

Specifies that the format of the unloaded segment records is same as IMS DBT 2.x. This is the system default value.

TFMT

Specifies that the format of the unloaded segment records is enhanced, which means that the prefix part of the record is generated based on the maximum number of segment levels defined in DBD.

AREA_INFORMATION_RECORD= or AIR=

Specifies whether to generate an area information record in an unloaded segment file.

Y | YES

Generates an area information record. This is the system default value.

N | NO

Does not generate an area information record.

LRECL=

Specifies the LRECL of the unloaded segment records file.

SEGTFMT

LRECL is determined based on the TFMT format prefix and the maximum length of segments defined in DBD. This value is effective only when FORMAT=TFMT is specified. FABCOP6M macro will cause FABD3676E message to be generated and ends with return code of 8 when FORMAT=TFMT is not specified.

BLOCK

LRECL is determined as BLKSIZE -4. This is the system default value.

OUTDD=

Specifies whether the DD statements for the output data sets are mandatory.

Y | YES

Specifies that the DD statement for output data sets, which is DURDzzzO, DURSzzzO, or UR6DBDFN, is required. This is the system default value.

N | NO

Specifies that the DD statement for output data sets, which is DURDzzzO, DURSzzzO, or UR6DBDFN, is optional.

Examples for the DEDB Reload Segment Data Set Create utility

These are example JCL statements for the DEDB Reload Segment Data Set Create utility.

FABCUR6 and FABCUR7 can be used to change the DEDB hierarchical structure.

When the FPA Unload function or the FPB Unload utility unloads the DEDB, the hierarchical structure is unchanged unless a NEWACB DD is specified. The hierarchical structure of the unloaded data set can be changed using FABCUR6 and FABCUR7.

To do this, use the following guidelines:

1. The DEDB unloaded file must be sorted before processing.
2. The ACBLIB data set for FABCUR6 must contain the new DMB defined for the DEDB being processed.
3. The application program reads each unloaded segment record into an I/O area using FABCUR7, and then puts the contents of the I/O area to the output file using FABCUR6.

The following figure shows example JCL stream used to change a DEDB hierarchical structure.

```
//CONVERT EXEC PGM=user-pgm
//UR7PRINT DD SYSOUT=A
//UR7AUDIT DD SYSOUT=A
//UR7DBDFN DD DSN=HPFP.UR.DURDBDFN,DISP=SHR
//UR7DATA DD DSN=HPFP.UR.FILEzzz.SORTED.SEGDATA,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//RMDLIB DD DSN=IMSVS.PGMLIB,DISP=SHR
//UR6PRINT DD SYSOUT=A
//UR6AUDIT DD SYSOUT=A
//UR6DBDFN DD HPFP.UR6.DURDBDFN,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(1,1))
//DURDzzzO DD DSN=HPFP.UR.FILEzzz.SEGDATA,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=TAPE,
//          DCB=BLKSIZE=20000
//DURSzzzO DD DSN=HPFP.UR.FILEzzz.SORTCARD,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(1,1))
```

Figure 25. Example of changing a DEDB hierarchical structure

The following figure shows example JCL stream used when using COBOL statements for FABCUR6 and FABCUR7.

```

IDENTIFICATION DIVISION.
PROGRAM_ID. CALL6510
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING_STORAGE SECTION.

*-----*
*  FABCUR7 DC & DS AREA
*-----*

01  FUNC_INIT      PIC X(04) VALUE 'INIT'.
01  FUNC_INID      PIC X(04) VALUE 'INID'.
01  FUNC_GET       PIC X(04) VALUE 'GET '.
01  FUNC_GET1      PIC X(04) VALUE 'GET1'.
01  FUNC_GET2      PIC X(04) VALUE 'GET2'.
01  FUNC_EOF       PIC X(04) VALUE 'EOF '.

*
01  STATUS_CODE    PIC X(02) VALUE ' '.
01  IO_AREA.
    05  GSEG_NAME      PIC X(08).
    05  GSEG_SSPTS     PIC X          OCCURS 8 TIMES.
    05  GSEG_DATA.
        07  GSEG_DATA_LL      PIC S9(4)    COMP.
        07  GSEG_DATA_CONTAIN PIC X(902).

*
01  STATUS_OK      PIC X(02) VALUE ' '.
01  STATUS_EOF     PIC X(02) VALUE 'GB'.

*-----*
*  FABCUR6 DC & DS AREA
*-----*

01  FUNC_PUT       PIC X(04) VALUE 'PUT '.
*
01  UR6_DBNAME     PIC X(08) VALUE 'DEDBJN22'.
*
01  RC_NORMAL      PIC 9(04) VALUE 0.

*
PROCEDURE DIVISION.
*
    PERFORM MAIN_RTN THRU MAIN_RTN_END.
    GOBACK.

```

Figure 26. COBOL statements for using FABCUR6 and FABCUR7 (Part 1 of 2)

```

* ----- *
* ----- MAIN ROUTINE ----- *
* ----- *
MAIN_RTN.
    PERFORM INIT_PROC THRU INIT_PROC_END.
    PERFORM GET_PROC THRU GET_PROC_END.
    PERFORM EOF_PROC THRU EOF_PROC_END.
MAIN_RTN_END. EXIT.

* ----- *
* ----- INITIALIZATION PROCESSING ----- *
* ----- *
INIT_PROC.
    CALL 'FABCUR7'          USING FUNC_INIT.
    CALL 'FABCUR6'          USING FUNC_INIT,
                             UR6_DBDDNAME.
INIT_PROC_END. EXIT.

* ----- *
* ----- GET PROCESSING ----- *
* ----- *
GET_PROC.
    PERFORM UNTIL STATUS_CODE = STATUS_EOF
        MOVE ZERO TO IO_AREA
        CALL 'FABCUR7'          USING FUNC_GET,
                                   STATUS_CODE,
                                   IO_AREA
        IF STATUS_CODE NOT = STATUS_EOF
            CALL 'FABCUR6'      USING FUNC_PUT,
                                   IO_AREA
        END_IF
    END_PERFORM.
GET_PROC_END. EXIT.

* ----- *
* ----- TERMINATION PROCESSING ----- *
* ----- *
EOF_PROC.
    CALL 'FABCUR7'          USING FUNC_EOF.
    CALL 'FABCUR6'          USING FUNC_EOF.
    MOVE RC_NORMAL TO RETURN_CODE.
EOF_PROC_END. EXIT.

```

Figure 27. COBOL statements for using FABCUR6 and FABCUR7 (Part 2 of 2)

Chapter 5. DEDB Unloaded Segment Data Set Retrieve utility

Use the DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) to retrieve the DEDB segment data from the DEDB unloaded segment data set created by the FPA Unload function and the FPB Unload utility.

Topics:

- [“Functions of the DEDB Unloaded Segment Data Set Retrieve utility” on page 53](#)
- [“Data and system flow of the DEDB Unloaded Segment Data Set Retrieve utility” on page 53](#)
- [“Calling the DEDB Unloaded Segment Data Set Retrieve utility \(from your program\)” on page 54](#)
- [“DD statements for the DEDB Unloaded Segment Data Set Retrieve utility” on page 58](#)
- [“Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61](#)
- [“Output for the DEDB Unloaded Segment Data Set Retrieve utility” on page 63](#)
- [“Examples for the DEDB Unloaded Segment Data Set Retrieve utility” on page 65](#)

Functions of the DEDB Unloaded Segment Data Set Retrieve utility

The DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) is called from a user application program. FABCUR7 retrieves the DEDB segment data from the DEDB unloaded segment data set created by DEDB Unload. A user program can then read two DEDB unloaded segment data sets from the same DEDB using the dual-mode processing feature.

The user application program does not need to handle:

- The format of the segment record.
- Reading the segment records from an input data set.

FABCUR7 is link-edited into a user program or can be invoked dynamically using ATTACH, LINK, or DYNAMIC CALLs. The IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0) must be concatenated to your application program library in the JOBLIB/STEPLIB DD statement.

Data and system flow of the DEDB Unloaded Segment Data Set Retrieve utility

This topic describes the data and system flow of the DEDB Reload Segment Data Set Retrieve utility.

ACB definitions are retrieved from the database definition record data set specified by the UR7DBDFN DD statement or from the ACB library specified by the ACBLIB DD statement. However, if the IMS catalog and the IMS management of ACBs are enabled, ACB definitions can be retrieved from the IMS directory.

The following figure shows the flow of FABCUR7.

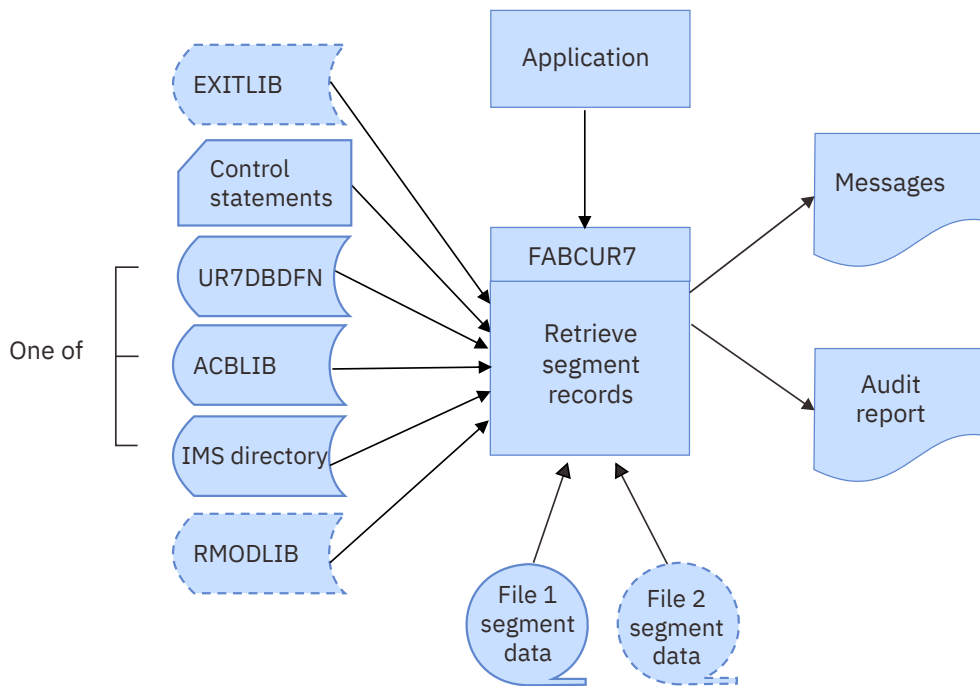


Figure 28. Flow of DEDB Unloaded Segment Data Set Retrieve utility

Calling the DEDB Unloaded Segment Data Set Retrieve utility (from your program)

FABCUR7 can be called from programs written in any language conforming to the z/OS register linkage conventions.

Example

The following figure shows example JCL for FABCUR7.

```
//UR7 EXEC PGM=appl-name,REGION=rrrrM
// DD dd statements necessary for an application
//EXITLIB DD DSN=USER.PGMLIB,DISP=SHR
//UR7DBDFN DD DSN=HPFP.UR.DURDBDFN,DISP=OLD
//UR7DATA DD DSN=HPFP.UR.FILEzzz.SEGDATA,DISP=OLD
//UR7PRINT DD SYSOUT=A
//UR7AUDIT DD SYSOUT=A
//UR7CTL DD *
... control statements ...
/*
```

Figure 29. Example JCL for FABCUR7

Dynamic linkage to an application

FABCUR7 can be invoked dynamically via ATTACH, LINK, or DYNAMIC CALLS.

An application can also use the alias name FABEUR7. The IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0) must be concatenated to your application program library in the JOBLIB/STEPLIB DD statement.

Static linkage to an application

FABCUR7 can be invoked statically.

FABCUR7 must be included from the IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0) when you link edit the program that calls FABCUR7. Another entry point name, FABEUR7, can be used by your program to call FABCUR7.

Application interface

FABCUR7 can be called with a parameter list containing a function code and one or more other data items depending on the requesting function. The following table shows the function codes.

Function code	Description
INIT	Initialization (single mode)
INID	Initialization (dual mode)
GET	Get segment data (single mode)
GET1	Get segment data from the first DEDB Unloaded Segment Data Set (dual mode)
GET2	Get segment data from the second DEDB Unloaded Segment Data Set (dual mode)
EOF	End of file

Initialization functions: INIT/INID

These functions do the following:

- Edit and parse the control statements.
- Read a DURDBDFN record and generate the necessary control blocks.
- Open all required output data sets.
- Load the user exit routine, if necessary.
- Load and run a DEDB area open call (entry code 12) to segment edit/compression routines, if necessary.

Note: The INIT or INID function must be run prior to any other function call.

Interface parameter list

Function code

This is a mandatory parameter. It is a 4-byte field containing an address of a 4-byte function code field containing INIT or INID.

Use function code INIT to initialize the utility in single mode. Use function code INID to initialize the utility in dual mode.

Note: The high-order bit of the last address parameter must be set to 1. The bit can be checked to determine the end of the list.

Coding INIT or INID in a COBOL program

For *single mode*, define the following in WORKING_STORAGE:

```
01 FUNC_INIT PIC X(04) VALUE 'INIT'.
```

For *dual mode*, define the following in WORKING_STORAGE:

```
01 FUNC_INIT PIC X(04) VALUE 'INID'.
```

In the PROCEDURE DIVISION, add the following:

```
CALL 'FABCUR7' USING FUNC_INIT.
```

Get segment data: GET/GET1/GET2

These functions do the following:

- Read next sequential segment record.

- Run a segment expansion call (entry code 4) if the IMSCOMP=EXP is provided and the COMPRTN parameter is defined in the DBD to the segment.
- Run an EXP (expand) function call to an exit routine, if the EXITRTN control statement is provided.
- Return segment name and data and optionally additional segment information.

Interface parameter list

Function code

This is a mandatory parameter. It is a 4-byte field containing an address of a 4-byte function code field containing GET, GET1, or GET2. The function code is left-adjusted in the field. Function code GET is used to specify single mode. Function code GET1 and GET2 are used to specify dual mode.

Status code area

This is a mandatory parameter. It is a 4-byte field containing an address of a 2-byte status code field in which FABCUR7 will set a return status code.

One of following status codes will be returned:

blanks

GET completed normally

GB

End of file

Segment I/O area

This is a mandatory parameter. It is a 4-byte field containing the address of an I/O area of a data segment being retrieved by FABCUR7.

The following table shows the layout of the Segment I/O Area.

Table 10. Layout of segment I/O area

Field name	Assembly code definition	Description
GSEG_NAME	CL8	Segment name
GSEG_SSPTRS	8CL1	Subset pointer indicator
GSEG_DATA	0XLnnn	Segment data including the 2-byte length field
GSEG_DATA_LL	XL2	Segment data length

Notes:

- *nnn* is the length of the longest segment.
- The subset pointer indicator is eight one-byte positional flags set left to right for pointers 1 - 8. They specify whether or not this segment is the target of a subset pointer. Setting Y in one of the flag bytes specifies that the segment is pointed to by the corresponding subset pointer.

Extended root/Segment information area

This is an optional parameter. It is a 4-byte field containing the address of an extended root/segment information area in which FABCUR7 will return necessary information.

The following table shows the layout of the extended root segment information area.

Table 11. Extended root segment information area

Field name	Assembly code definition	Description
E_RAP_RBA	F	RAP RBA
E_ROOT_SEQ	PL4	Root sequence
	F	Reserved

Table 11. Extended root segment information area (continued)

Field name	Assembly code definition	Description
E_R_KEY_LL	H	Root key length
E_R_KEY_DATA	XL256	Root key data
E_CS_TYPE	CL4	Current segment type: ROOT, DDEP, or SDEP
E_CS_CODE	H	Current segment code
E_CS_H_LVL	H	Current segment hierarchical level
E_CS_KEY_LL	H	Current segment key length
E_CS_KEY_DATA	XL256	Current segment key data

Note: The high-order bit of the last address parameter must be set to 1. The bit can be checked to determine the end of the list.

Coding GET, GET1, or GET2 in a COBOL program

For *single mode*, define the following in WORKING_STORAGE:

```
01 FUNC_GET          PIC X(04)    VALUE 'GET '.
01 STATUS_CODE       PIC X(02)    VALUE ' '.
01 IO_AREA.
   05 SEG_NAME        PIC X(08).
   05 SEG_SSPTS       PIC X          OCCURS 8 TIMES.
   05 SEG_DATA        PIC X(nnn).
   05 PSEG_DATA       REDEFINES SEGDATA.
   07 PSEG_DATA_LL    PIC S9(4)    COMP.
   07 PSEG_DATA_CONTAIN PIC X(nnn-2).
```

where *nnn* is the length of the longest segment defined in DBDGEN.

In the PROCEDURE DIVISION, add the following:

```
CALL 'FABCUR7' USING FUNC_GET,
STATUS_CODE,
IO_AREA.
```

To retrieve extended root/segment information, also define the following in WORKING_STORAGE:

```
01 EXTENDED_IO_AREA.
   05 E_RAP_AREA      PIC S9(8) COMP.

   05 E_ROOT_SEQ      PIC S9(7) COMP-3.

   05 E_FILLER        PIC S9(8) COMP.
   05 E_R_KEY_LL      PIC S9(4) COMP.
   05 E_R_KEY_DATA    PIC X(256).
   05 E_CS_TYPE       PIC X(4).
   05 E_CS_CODE       PIC S9(4) COMP.
   05 E_CS_H_LVL      PIC S9(4) COMP.
   05 E_CS_KEY_LL     PIC S9(4) COMP.
   05 E_CS_KEY_DATA   PIC X(256).
```

In the PROCEDURE-DIVISION, add the following:

```
CALL 'FABCUR7' USING FUNC_GET,
STATUS_CODE,
IO_AREA,
EXTENDED_IO_AREA.
```

For *dual mode*, define the following in WORKING_STORAGE:

```
01 FUNC_GET1          PIC X(04) VALUE 'GET1 ' .
01 FUNC_GET2          PIC X(04) VALUE 'GET2' .
01 STATUS_CODE        PIC X(02) VALUE ' ' .
01 IO_AREA.
05 SEG_NAME           PIC X(08) .
05 SEG_SSPTS          PIC X OCCURS 8 TIMES .
05 SEG_DATA           PIC X(nnn) .
05 PSEG_DATA          REDEFINES SEGDATA .
07 PSEG_DATA_LL       PIC S9(4) COMP .
07 PSEG_DATA_CONTAIN  PIC X(nnn-2) .
```

where *nnn* is the length of the longest segment defined in DBDGEN.

In the PROCEDURE DIVISION, add the following:

```
CALL 'FABCUR7' USING FUNC_GET1,
                     STATUS_CODE,
                     IO_AREA.
                     ** PROCESS **
CALL 'FABCUR7' USING FUNC_GET2,
                     STATUS_CODE,
                     IO_AREA.
```

End of file: EOF

This function is invoked after all segments for a database have been processed. This function does the following:

- Generates the audit report.
- Runs a DEDB area close call (entry code 16) to the segment edit/compression routines, if necessary.
- Runs an END (clean up) function call to an exit routine if the EXITRTN control statement is provided.
- Closes all output data sets.

Interface parameter list

Function code

This is a mandatory parameter. It is a 4-byte field containing an address of the 4-byte function code field containing EOF, left-adjusted in the field.

Note: The high-order bit of the last address parameter must be set to 1. The bit can be checked to determine the end of the list.

Coding EOF in a COBOL program

In WORKING_STORAGE, define the following:

```
01 FUNC_EOF PIC X(04) VALUE 'EOF ' .
```

In the PROCEDURE DIVISION, add the following:

```
CALL 'FABCUR7' USING FUNC_EOF.
```

DD statements for the DEDB Unloaded Segment Data Set Retrieve utility

DD statements for the DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) determine the input and output data sets and how the utility is run.

The following table shows the JCL DD statements required to be included in the job step.

Table 12. FABCUR7 DD statements

DDNAME	Use	Format	Required or optional
JOBLIB/STEPLIB	Input	PDS	Required
UR7PRINT	Output	LRECL=133	Required
UR7AUDIT	Output	LRECL=133	Required
ACBLIB	Input	PDS	Optional
IMSACBA	Input	PDS	Optional
IMSACBB	Input	PDS	Optional
MODSTAT	Input		Optional
MODSTAT2	Input		Optional
RMODLIB	Input	PDS	Optional
EXITLIB	Input	PDS	Optional
UR7CTL	Input	LRECL=80	Optional
UR7DBDFN	Input		Optional
UR7DATA	Input		Required
UR7DATA1	Input		Required
UR7DATA2	Input		Required

JOBLIB/STEPLIB DD

Defines the library that contains the user application program.

The following libraries must be concatenated to your application program library in the JOBLIB/STEPLIB DD statement:

- The IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0).
- The IMS Tools Base library (SGLXLOAD), if you specify the IMSCATHLQ=*bsdshlq* keyword.

UR7PRINT DD

Defines the output data set that contains messages issued by FABCUR7. The data set can reside on a direct-access device or printer, or be routed through the output stream. You can code RECFM=FBA, LRECL=133 on your DD statement, but it is better to use:

```
//UR7PRINT DD SYSOUT=A
```

UR7AUDIT DD

Defines the output data set that contains the FABCUR7 Audit Control report. The data set can reside on a direct-access device or printer, or be routed through the output stream. You can code RECFM=FBA, LRECL=133 on your DD statement, but it is better to use:

```
//UR7AUDIT DD SYSOUT=A
```

ACBLIB DD

Defines the library that contains the DMB for the database.

If MODSTAT/MODSTAT2 DD is provided, this DD is not necessary.

You must supply this DD statement if you do not specify the IMSCATHLQ=*bsdshlq* statement or the UR7DBDFN DD statement.

If IMSCATHLQ=*bsdshlq* is present, the utility ignores the ACBLIB DD statement. The IMS directory is used instead of the ACB library.

IMSACBA DD

Defines the library that contains the DMB for the database. If MODSTAT/MODSTAT2 DD is specified and if UR7DBDFN DD is not specified, this DD must be provided.

IMSACBB DD

Defines the library that contains the DMB for the database. If MODSTAT/MODSTAT2 DD is specified and if UR7DBDFN DD is not specified, this DD must be provided.

MODSTAT DD

Defines the MODSTAT data set. If this DD is specified, IMSACBA and IMSACBB DD must be specified instead of the ACBLIB DD.

MODSTAT2 DD

Defines the MODSTAT2 data set. If this DD is specified, IMSACBA and IMSACBB DD must be specified instead of the ACBLIB DD.

RMODLIB DD

Defines the library that contains the segment edit/compression routine. This statement is required if there are compressed segments and the EXITRTN control statement or the IMSCOMP=EXP control statement is specified.

If this DD statement is not provided, an attempt is made to load the edit/compression routine from JOBLIB/STEPLIB.

EXITLIB DD

Defines the library that contains the program load module specified in the EXITRTN control statement.

UR7CTL DD

Defines the control statement input data set. This data set can reside on a direct-access device, or be routed through the input stream.

UR7DBDFN DD

Defines a data set that contains a formatted copy of a DMB. It is the DURDBDFN data set from FABCUR1 or that generated by FABCUR5.

Instead of UR7DBDFN DD, you can specify ACBLIB DD, which defines the ACB library that contains the DMB.

If you want to use the IMS directory instead of the UR7DBDFN data set, specify IMSCATHLQ=*bsdshlq*, and do not specify the UR7DBDFN DD statement.

UR7DATA DD

Defines the input data set that contains the unloaded (or created) segment data records for one or more areas. This DD statement is required for single mode.

UR7DATA1 DD

Defines the input data set that contains the unloaded (or created) segment data records for one or more areas. This DD statement is required for dual-mode operation.

UR7DATA2 DD

Defines the input data set that contains the unloaded (or created) segment data records for one or more areas. This DD statement is required for dual-mode operation.

Input for the DEDB Unloaded Segment Data Set Retrieve utility

You must specify DD statements for the job control language (JCL) to run the DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7).

UR7DATA, UR7DATA1, and UR7DATA2 DD data sets

The UR7DATA, UR7DATA1, and UR7DATA2 data sets contain the DEDB segment data records to be processed by FABCUR7.

DEDB segment data records in the DEDB unloaded segment data set must be sorted at least by their root key values and the values in their sequence fields. That is, in the data set, all segments that belong to the same DEDB record must be contiguous in the IMS DL/I GN call hierarchical sequence.

UR7CTL DD data set

The UR7CTL data set contains the user's description to run the processes by FABCUR7 to retrieve unloaded segment records.

Format

This control statement data set usually resides in the input stream. However, it can also be defined as a sequential data set or as a member of a partitioned data set. It must contain 80-byte, fixed-length records. Block size, if coded, must be a multiple of 80. This data set can contain several different types of control statement, including a comment statement. It is coded as shown in the following figure.

```
//UR7CTL DD *  
EXITRTN=exit-routine  
/*
```

Figure 30. FABCUR7 UR7CTL data set

Control statements

The FABCUR7 control statements are:

- DBDNAME
- EXITRTN
- IMSCOMP
- AREA_INFORMATION_RECORD
- IMSCATHLQ
- IMSCATACB_INPUT

The control statements are read in from the input source specified in the UR7CTL DD statement.

For information about the syntax of these control statements, see in the topic "FILECTL control statement" (for DEDB Unload) in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

DBDNAME control statement

The DBDNAME statement specifies the DBD name of the DEDB for the unloaded segment records file. There must be only one DBDNAME statement. This statement is required if ACBLIB DD is specified instead of UR7DBDFN DD. If UR7DBDFN DD is specified, this statement is not necessary. If DBDNAME and UR7DBDFN DD is specified, the UR7DBDFN file is validated to make sure that the correct DURDBDFN file is provided.

```
DBDNAME=dbdname
```

DBDNAME=

This statement specifies the DEDB DBD name that is used for:

- Validating the UR7DBDFN file.
- Creating the Database Definition Record (DURDBDFN) internally on the basis of a DEDB DMB when ACBLIB DD is specified instead of UR7DBDFN DD.

DBDNAME is an optional keyword.

dbdname

Specifies the name of the DBD that is to be used.

EXITRTN control statement

The optional EXITRTN statement specifies the name of the user exit routine that will be invoked with an EXPAND function.

```
[EXITRTN=exit-routine]
```

EXITRTN=

This optional keyword specifies the name of the user exit routine to be invoked with an EXPAND function.

exit-routine

Identifies the name of the user exit routine that will be called.

An EXITLIB DD statement must be provided when this control statement is specified.

For more information about the exit routine, see the topic "Exit routine option and its interface" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

IMSCOMP control statement

The optional IMSCOMP statement specifies whether the segment edit/compression routine will be invoked with a segment expansion call (entry code 4) for candidate segments.

```
[IMSCOMP={NO|EXP}]
```

IMSCOMP=

This optional keyword determines whether the segment edit/compression routine will be invoked with a segment expansion call (entry code 4) for a candidate segment.

NO

Means that the segment edit/compression routine is not invoked. This is the default value.

EXP

Means that the segment edit/compression routine is invoked with a segment expansion call (entry code 4) for candidate segments

Note: Even if IMSCOMP=NO is specified, the segment edit/compression routine is invoked to expand the compressed segment before calling the user exit routine when the EXITRTN control statement is specified. The expanded segment is compressed again after calling the exit routine and the compressed segment is returned to the application program.

AREA_INFORMATION_RECORD control statement

The optional AREA_INFORMATION_RECORD statement specifies whether the input unloaded segment records file on a UR7DATA DD statement contains the area information records.

```
AREA_INFORMATION_RECORD={YES|NO}
```

AREA_INFORMATION_RECORD=

The optional AREA_INFORMATION_RECORD statement specifies whether the input unloaded segment records file on a UR7DATA DD statement contains the area information records.

YES

Specifies that area information records should be contained in the UR7DATA DD data set. This is the default value.

NO

Specifies that no area information records should be contained in the UR7DATA DD data set.

An abbreviation AIR is used for AREA_INFORMATION_RECORD.

IMSCATHLQ control statement

The optional IMSCATHLQ statement specifies the high-level qualifier of the bootstrap data set of the IMS directory, which is an extension of the IMS catalog. You must enable the IMS catalog and the IMS management of ACBs when you specify the high-level qualifier of the bootstrap data set of the IMS directory.

```
IMSCATHLQ={*NO|bsdshlq}
```

IMSCATHLQ=

This control statement has the following keywords:

bsdshlq

Reads the ACB member from the IMS directory instead of the ACB library by using IMS Tools Catalog Interface. *bsdshlq* specifies the high-level qualifier of the IMS directory bootstrap data set.

***NO**

Reads the ACB member from the ACB library. IMSCATHLQ=*NO is the default value.

IMSCATACB_INPUT control statement

The optional IMSCATACB_INPUT statement specifies whether to retrieve the currently active ACB definition or the pending ACB definition from the IMS directory. This statement is effective only when the IMSCATHLQ=*bsdshlq* statement is specified.

```
IMSCATACB_INPUT={CURRENT|PENDING}
```

IMSCATACB_INPUT=

This control statement has the following keywords:

CURRENT

The currently active ACB member is retrieved from the IMS directory data sets. IMSCATACB_INPUT=CURRENT is the default value.

PENDING

The pending ACB member is retrieved from the staging data set.

Output for the DEDB Unloaded Segment Data Set Retrieve utility

The following topics describe the output produced by FABCUR7.

UR7PRINT DD data set

The UR7PRINT data set contains the messages issued by the FABCUR7 program. This data set contains 133-byte records.

Format

If you code the block size in your JCL, it must be a multiple of 133. It is better to code your DD statement as follows:

```
//UR7PRINT DD SYSOUT=A
```

FABCUR7-Messages report

The following figure shows an example of the Messages report.

FABC0700I - FABCUR7 ENDED NORMALLY

Figure 31. FABCUR7-Messages report

UR7AUDIT DD data set

The DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) generates one Audit Control report to provide verification totals.

Format

If you code the block size in your JCL, it must be a multiple of 133. It is better to code your DD statement as follows:

```
//UR7AUDIT DD SYSOUT=A
```

FABCUR7 Audit Control report-Segment totals by input file

This report shows segment counts and area totals by the input file ddname. File totals and a database total are also provided. The following figure shows an example of the Audit Control report.

SEGMENT TOTALS BY INPUT FILE:

FILE DDNAME	AREA NO	SEG CODE	SEG NAME	NUMBER SEGMENTS	
UR7DATA	1	1	ROOTSEG1	2	
		2	SDSEGNM1	0	
		3	DD1	6	
		4	DD2	6	
		5	DD3	7	
		6	DD4	4	
		7	DD43	2	
		8	DD44	0	
		9	DD5	1	
		10	DD53	0	

	**	AREA TOTALS	**		28

	2	1	ROOTSEG1	0	
		2	SDSEGNM1	0	
		3	DD1	0	
		4	DD2	0	
		5	DD3	0	
		6	DD4	0	
		7	DD43	0	
		8	DD44	0	
		9	DD5	0	
		10	DD53	0	

	**	AREA TOTALS	**		0

	3	1	ROOTSEG1	0	
		2	SDSEGNM1	0	
		3	DD1	0	
		4	DD2	0	
		5	DD3	0	
		6	DD4	0	
		7	DD43	0	
		8	DD44	0	
		9	DD5	0	
		10	DD53	0	

	**	AREA TOTALS	**		0

	4	1	ROOTSEG1	0	
		2	SDSEGNM1	0	
		3	DD1	0	
		4	DD2	0	
		5	DD3	0	
		6	DD4	0	
		7	DD43	0	
		8	DD44	0	
		9	DD5	0	
		10	DD53	0	

** AREA TOTALS			**	0
5	1	ROOTSEG1	0	
	2	SDSEGNM1	0	
	3	DD1	0	
	4	DD2	0	
	5	DD3	0	
	6	DD4	0	
	7	DD43	0	
	8	DD44	0	
	9	DD5	0	
	10	DD53	0	
** AREA TOTALS			**	0
*** FILE TOTALS			***	28
**** DATABASE TOTAL			****	28

Figure 32. FABCUR7 Audit Control report

Examples for the DEDB Unloaded Segment Data Set Retrieve utility

These are example JCL statements for the DEDB Unloaded Segment Data Set Retrieve utility.

The figure in this topic shows example JCL stream used when using COBOL statements for FABCUR6 and FABCUR7.

When the FPA Unload function or the FPB Unload utility unloads the DEDB, the hierarchical structure is unchanged unless a NEWACB DD is specified. The hierarchical structure of the unloaded data set can be changed using FABCUR6 and FABCUR7.

To do this, use the following guidelines:

1. The DEDB unloaded file must be sorted before processing.
2. The ACBLIB data set for FABCUR6 must contain the new DMB defined for the DEDB being processed.
3. The application program reads each unloaded segment record into an I/O area using FABCUR7, and then puts the contents of the I/O area to the output file using FABCUR6.

The following figure shows example JCL stream used to change a DEDB hierarchical structure.

```
//CONVERT EXEC PGM=user-pgm
//UR7PRINT DD SYSOUT=A
//UR7AUDIT DD SYSOUT=A
//UR7DBDFN DD DSN=HPFP.UR.DURDBDFN,DISP=SHR
//UR7DATA DD DSN=HPFP.UR.FILEzzz.SORTED.SEGDATA,DISP=SHR
//ACBLIB DD DSN=IMSVS.ACBLIB,DISP=SHR
//RMDLIB DD DSN=IMSVS.PGMLIB,DISP=SHR
//UR6PRINT DD SYSOUT=A
//UR6AUDIT DD SYSOUT=A
//UR6DBDFN DD HPFP.UR6.DURDBDFN,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(1,1))
//DURDzzz0 DD DSN=HPFP.UR.FILEzzz.SEGDATA,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=TAPE,
//          DCB=BLKSIZE=20000
//DURSzzz0 DD DSN=HPFP.UR.FILEzzz.SORTCARD,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(1,1))
```

Figure 33. Example of changing a DEDB hierarchical structure

The following figure shows example JCL stream used when using COBOL statements for FABCUR6 and FABCUR7.

```

IDENTIFICATION DIVISION.
PROGRAM_ID. CALL6510
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING_STORAGE SECTION.

*-----*
*   FABCUR7 DC & DS AREA
*-----*

01  FUNC_INIT      PIC X(04) VALUE 'INIT'.
01  FUNC_INID      PIC X(04) VALUE 'INID'.
01  FUNC_GET       PIC X(04) VALUE 'GET '.
01  FUNC_GET1      PIC X(04) VALUE 'GET1'.
01  FUNC_GET2      PIC X(04) VALUE 'GET2'.
01  FUNC_EOF       PIC X(04) VALUE 'EOF '.

*
01  STATUS_CODE    PIC X(02) VALUE ' '.
01  IO_AREA.
    05  GSEG_NAME          PIC X(08).
    05  GSEG_SSPTS        PIC X          OCCURS 8 TIMES.
    05  GSEG_DATA.
        07  GSEG_DATA_LL      PIC S9(4)    COMP.
        07  GSEG_DATA_CONTAIN PIC X(902).

*
01  STATUS_OK      PIC X(02) VALUE ' '.
01  STATUS_EOF     PIC X(02) VALUE 'GB'.

*-----*
*   FABCUR6 DC & DS AREA
*-----*

01  FUNC_PUT       PIC X(04) VALUE 'PUT '.

*
01  UR6_DBNAME     PIC X(08) VALUE 'DEDBJN22'.

*
01  RC_NORMAL      PIC 9(04) VALUE 0.

*
PROCEDURE DIVISION.
*
    PERFORM MAIN_RTN THRU MAIN_RTN_END.
    GOBACK.

```

Figure 34. COBOL statements for using FABCUR6 and FABCUR7 (Part 1 of 2)

```

* -----*
* -----MAIN ROUTINE-----*
* -----*
MAIN_RTN.
    PERFORM INIT_PROC THRU INIT_PROC_END.
    PERFORM GET_PROC THRU GET_PROC_END.
    PERFORM EOF_PROC THRU EOF_PROC_END.
MAIN_RTN_END. EXIT.

* -----*
* -----INITIALIZATION PROCESSING-----*
* -----*
INIT_PROC.
    CALL 'FABCUR7'          USING FUNC_INIT.
    CALL 'FABCUR6'          USING FUNC_INIT,
                             UR6_DBNAME.
INIT_PROC_END. EXIT.

* -----*
* -----GET PROCESSING-----*
* -----*
GET_PROC.
    PERFORM UNTIL STATUS_CODE = STATUS_EOF
        MOVE ZERO TO IO_AREA
        CALL 'FABCUR7'          USING FUNC_GET,
                                   STATUS_CODE,
                                   IO_AREA
        IF STATUS_CODE NOT = STATUS_EOF
            CALL 'FABCUR6'      USING FUNC_PUT,
                                   IO_AREA
        END_IF
    END_PERFORM.
GET_PROC_END. EXIT.

* -----*
* -----TERMINATION PROCESSING-----*
* -----*
EOF_PROC.
    CALL 'FABCUR7'          USING FUNC_EOF.
    CALL 'FABCUR6'          USING FUNC_EOF.
    MOVE RC_NORMAL TO RETURN_CODE.
EOF_PROC_END. EXIT.

```

Figure 35. COBOL statements for using FABCUR6 and FABCUR7 (Part 2 of 2)

Chapter 6. HD To DEDB Unload Data Set Conversion utility

Use the HD To DEDB Unload Data Set Conversion utility (FABCUR8) to convert an HD unload data set to a DEDB Unloaded segment data set.

Topics:

- [“Functions of the HD To DEDB Unload Data Set Conversion utility” on page 69](#)
- [“Data and system flow of the HD To DEDB Unload Data Set Conversion utility” on page 70](#)
- [“Running the HD To DEDB Unload Data Set Conversion utility” on page 70](#)
- [“DD statements for the HD To DEDB Unload Data Set Conversion utility” on page 71](#)
- [“Input for the HD To DEDB Unload Data Set Conversion utility” on page 74](#)
- [“Output for the HD To DEDB Unload Data Set Conversion utility” on page 78](#)
- [“Example of the HD To DEDB Unload Data Set Conversion utility” on page 79](#)

Functions of the HD To DEDB Unload Data Set Conversion utility

The function of FABCUR8 is to convert an HD unload data set to a DEDB Unloaded segment data set.

The following types of database that are unloaded by the IMS HD Reorganization Unload Utility (DFSURGU0), IMS High Performance Unload (FABHURG1), or that are created by any other programs that conform to the format of the IMS standard HD unload record are supported:

- HDAM database
- HIDAM database
- HISAM database
- PHDAM database
- PHIDAM database

PSINDEX database is not supported.

The utility not only converts an HD unload data set to the format of the DEDB database of the same structure except logical segments, but also to a new DEDB database structure. In the new structure, you can define new segments so far as all physical segments in the original HD database are kept in the same hierarchical structural order.

FABCUR8 calls FABCUR6 internally to create a DEDB unloaded segment record data set.

Restriction:

An HD unload data set of any of following conditions is not supported:

- HD unload data set for PSINDEX database.
- A physical segment in the HD DBD that is not defined in the DEDB DMB.
- A physical segment that is defined as fixed length in HD DBD.
- A physical segment in the HD DBD that is defined as fixed length in DEDB DMB.
- A physical segment in the HD DBD that is defined with edit/compression attribute. FABCUR8 will produce a DEDB unload segment data set with incorrect segment data when such HD unload data set is specified without providing HD DBD name in the EXEC parameter.
- All physical segments in the HD DBD are defined in the DEDB DMB but not in the same hierarchical structure order.

Data and system flow of the HD To DEDB Unload Data Set Conversion utility

This topic describes the data and system flow of the HD To DEDB Unload Data Set Conversion utility.

The following figure shows the flow of FABCUR8.

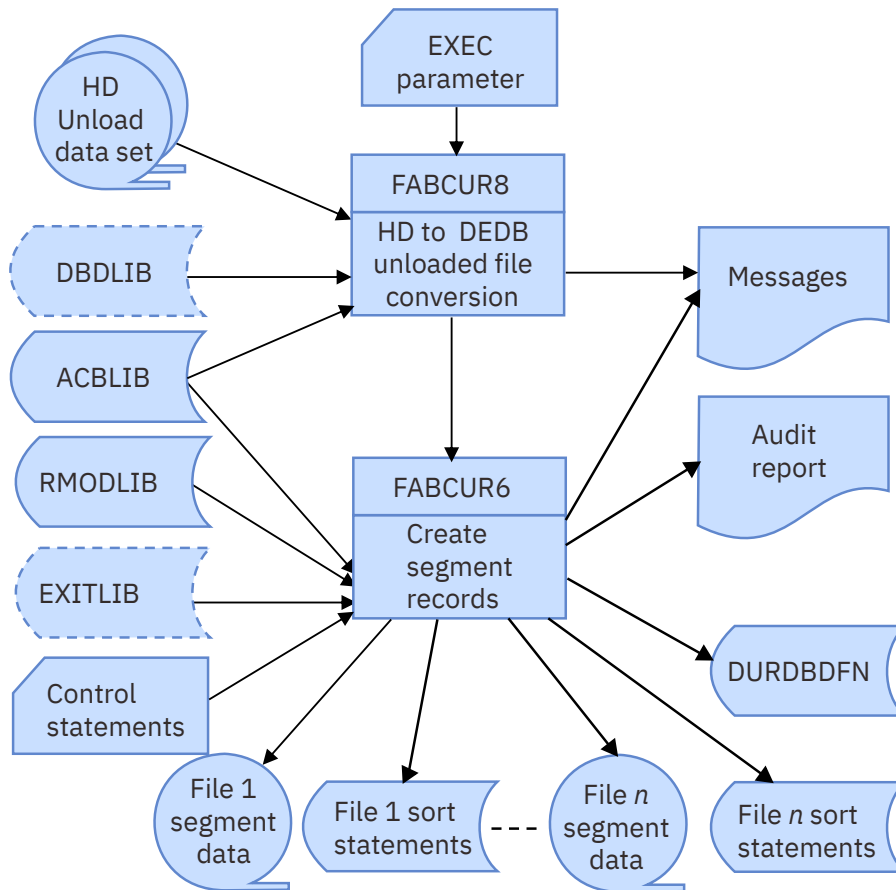


Figure 36. Flow of HD To DEDB Unload Data Set Conversion utility

Running the HD To DEDB Unload Data Set Conversion utility

The HD To DEDB Unload Data Set Conversion utility (FABCUR8) is run as a standard z/OS batch job. An EXEC statement and DD statements that define the input and output data sets are required. FABCUR8 has the alias name FABEUR8.

Procedure

1. Code the JCL for the FABCUR8 job step.
2. Specify the DD statements to define input data sets, output data sets, and how the function is run.
3. Run the JCL.

Example

The following figure shows example JCL. You can invoke FABCUR8 by using the alias as follows:

```
// EXEC PGM=FABEUR8,PARM='aaaaaaaa,bbbbbbbb',REGION=rmm
```

DD statements for the HD To DEDB Unload Data Set Conversion utility

DD statements for the HD To DEDB Unload Data Set Conversion utility (FABCUR8) determine the input and output data sets and how the utility is run.

The following table summarizes the DD statements for FABCUR8.

Table 13. FABCUR8 DD statements

DDNAME	Use	Format	Required or optional
JOBLIB or STEPLIB	Input	PDS	Required
ACBLIB	Input	PDS	Required
IMSACBA	Input	PDS	Optional
IMSACBB	Input	PDS	Optional
MODSTAT	Input		Optional
MODSTAT2	Input		Optional
DBDLIB	Input	PDS	Optional
RMODLIB	Input	PDS	Required
EXITLIB	Input	PDS	Optional
DURINPT	Input	LRECL=80	Required
UR6CTL	Input	LRECL=80	Optional
UR6PRINT	Output	LRECL=133	Required
UR6AUDIT	Output	LRECL=133	Required
UR6DBDFN	Output	Do no code DCB	Required
DURDzzzO or XDzzzzzO	Output	Do not code any DCB other than BLKSIZE	Required
DURDzzzE or XDzzzzzE	Output	Do not code any DCB other than BLKSIZE	Optional
DURSzzzO or XSzzzzzO	Output	LRECL=80	Required

All output data sets are blocked to the maximum size of the output device (unless overridden in the execution JCL). Because the blocking factor is determined at run time, standard labels must be used on all output data sets except for UR6PRINT and UR6AUDIT.

FABCUR8 calls FABCUR6 internally to create a DEDB unloaded segment record data set. Therefore DD statements for FABCUR6 are also needed.

EXEC

The EXEC statement must be in the form:

```
// EXEC PGM=FABCUR8,REGION=rrrrM,  
// PARM='aaaaaaaa,bbbbbbbb'
```

But FABCUR8 has the alias name FABEUR8, therefore you can invoke FABCUR8 by using the alias as follows:

```
// EXEC PGM=FABEUR8,REGION=rrrrM,  
// PARM='aaaaaaaa,bbbbbbbb'
```

aaaaaaaa

is the name of the DEDB DMB in ACBLIB.

bbbbbbbb

is the name of the full-function DBD in DBDLIB. This parameter is optional, but it is strictly recommended to specify this parameter in order to make sure that the input HD unload file and the target DEDB DMB are correct.

ACBLIB DD

Defines the ACB library that contains the DMB for the DEDB database. This DD statement is required. If MODSTAT/MODSTAT2 DD is provided, this DD is not necessary.

IMSACBA DD

Defines the ACB library that contains the DMB for the DEDB database. If MODSTAT/MODSTAT2 DD is specified, this DD statement is required.

IMSACBB DD

Defines the ACB library that contains the DMB for the DEDB database. If MODSTAT/MODSTAT2 DD is specified, this DD statement is required.

MODSTAT DD

Defines the MODSTAT data set. When this DD is specified, the IMSACBA and IMSACBB DD must be specified instead of the ACBLIB DD.

MODSTAT2 DD

Defines the MODSTAT2 data set. When this DD is specified, the IMSACBA and IMSACBB DD must be specified instead of the ACBLIB DD.

DBDLIB DD

Defines the DBD library that contains the DBD for the HD database. This DD statement is optional. This DD statement is required when HD DBD name is specified in the EXEC parameter. When this DD statement is not specified, DBD segment definition information will be obtained from the header record of the HD unloaded segment records file, but it is strictly recommended to specify the HD DBD name in the EXEC parameter and the DD statement in order to make sure that the input HD unload file and the target DEDB DMB are correct.

RMODLIB DD

Defines the library that contains the randomizing routine, the segment edit/compression routine, or both. Instead of defining the library on this DD statement, the library can be concatenated on the STEPLIB/JOBLIB DD statement.

EXITLIB DD

Defines the library that contains the program load module that is specified in the EXITRTN control statement. If EXITRTN control statement is specified, this DD statement is required.

DURINPT DD

Defines the input data that is produced by the IMS HD Reorganization Unload Utility (DFSURGU0), IMS High Performance Unload (FABHURG1), or that is created by any other programs that conform to the format of the IMS standard HD unload record. An HD unload file of PSINDEX is not allowed.

This DD statement is required.

Note: For HALDB database, if either a single HALDB partition or a range of HALDB partitions was unloaded, it is recommended that you concatenate HD unloaded files of all partitions.

UR6CTL DD

Defines the control statement input data set. This data set can reside on a direct-access device, or be routed through the input stream.

UR6PRINT DD

Defines the output data set that contains messages that are issued by FABCUR8 and FABCUR6. The data set can reside on a direct-access device or printer, or be routed through the output stream. You can code RECFM=FBA, LRECL=133 on your DD statement, but it is recommended that you use:

```
//UR6PRINT DD SYSOUT=A
```


UR6AUDIT DD

Defines the output data set that contains the FABCUR6 Audit Control report. The data set can reside on a direct-access device or printer, or be routed through the output stream. You can code RECFM=FBA, LRECL=133 on your DD statement, but it is recommended that you use:

```
//UR6AUDIT DD SYSOUT=A
```

UR6DBDFN DD

Defines an output data set for the database definition record that is generated by FABCUR6. This contains the data that is extracted from the DMB that is used by the reload processor. The data set must reside on a direct-access device. Space requirements depend on the size of the DMB, but a two tracks are usually enough. Do not code DCB information in your JCL.

Do not specify DISP=MOD for this DD statement.

DURDzzzO or XDzzzzzO DD

Defines an output data set for all of the database segment records that are produced for one or more of the areas defined in the DMB. A DURDzzzO DD statement is for areas in the range of 1 - 999, and an XDzzzzzO DD statement is for areas in the range of 1 - 2048. If the area number of the unloaded area is greater than 999, you should provide the XDzzzzzO DD statement. The value of zzz or zzzzz is made up of right-aligned digits, with leading zeros if needed.

DCB attributes are calculated by FABCUR6. RECFM is VB, and the default block size is the maximum block size of the output device. You can specify to override a block size on the DD statement. Do not code any other DCB parameters in your JCL.

Do not specify DISP=MOD for these DD statements.

The rules for supplying the DURDzzzO or the XDzzzzzO data sets are discussed in the description of the FILECTL control statement in [“UR6CTL DD data set” on page 74](#).

DURDzzzE or XDzzzzzE DD

Defines the second copy data set for DURDzzzO or XDzzzzzO. For a DURDzzzE or XDzzzzzE DD statement, there must be a corresponding DURDzzzO or XDzzzzzO DD statement. A DURDzzzE DD statement is for areas in the range of 1 - 999, and an XDzzzzzE DD statement is for areas in the range of 1 - 2048. If the area number of the unloaded area is greater than 999, you should provide the XDzzzzzE DD statement. The value of zzz or zzzzz is made up of right-aligned digits, with leading zeros if needed.

DCB attributes are calculated by FABCUR6. RECFM is VB, and the default block size is the maximum block size of the output device. You can specify to override a block size on the DD statement. Do not code any other DCB parameters in your JCL.

Do not specify DISP=MOD for these DD statements.

DURSzzzO or XSzzzzzO DD

Defines an output data set that contains the SORT control statements for the segment data set that is associated with it. There must be a DURSzzzO or XSzzzzzO data set for each DURDzzzO or XDzzzzzO data set. The data set must reside on a direct-access device. This data set is required even if the corresponding DURDzzzO or XDzzzzzO data set does not need to be sorted. A DURSzzzO DD statement is for areas in the range of 1 - 999, and an XSzzzzzO DD statement is for areas in the range of 1 - 2048. If the area number of the unloaded area is greater than 999, you should provide the XSzzzzzO DD statement. The value of zzz or zzzzz is made up of right-aligned digits, with leading zeros if needed. The space required is very small; one track suffices.

The DCB information is hard-coded in FABCUR6. Do not code the DCB information in your JCL.

Do not specify DISP=MOD for these DD statements.

Input for the HD To DEDB Unload Data Set Conversion utility

You must specify the necessary input DD data sets to run FABCUR8.

UR6CTL DD data set

The UR6CTL data set contains the user's description of the creating processes for reloading segment records to be done by FABCUR6.

Format

This control statement data set usually resides in the input stream. However, it can also be defined as a sequential data set or as a member of a partitioned data set. It must contain 80-byte, fixed-length records. Block size, if coded, must be a multiple of 80. This data set can contain several different types of control statements, including a comment statement. It can be coded as shown in the following figure.

```
//UR6CTL DD *  
FILECTL=1,ALL  
EXITRTN=exit-routine  
/*
```

Figure 37. FABCUR6 UR6CTL data set

Control statements

The FABCUR6 control statements are:

- FILECTL
- EXITRTN
- IMSCOMP
- USERCTL
- FORMAT
- AREA_INFORMATION_RECORD
- LRECL
- OUTDD

The control statements are read in from the input source specified in the UR6CTL DD statement.

For information about the syntax of these control statements, see the topic "DEDB Unload SYSIN DD data set control statements" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

FILECTL control statement

The optional FILECTL statement controls grouping of multiple areas' segment data into a single output file.

```
[FILECTL={zzzzzz}, {ALL|x|(x,y,...)|(x-y)|(*)}]
```

FILECTL=

Here zzzzz is a decimal number 1 - 2048. This statement is composed of the following:

zzzzz

Specifies the output file number described by this control statement. The number is specified as a 1-to-5 digit decimal number. There must be related DD statements that have this number for an unloaded file, DURDzzzO and an output data set for the SORT control statement, DURSzzzO in the JCL stream for each data set specified on a FILECTL control statement. The value of zzz or zzzzz in the DD name is made up of right-aligned digits, with leading zeros if need.

ALL

Writes the segment data records for all output areas into the DURDzzzO data set.

x

Writes the segment data records for output area x into the DURDzzzO data set.

(x,y,...)

Writes the segment data records for output area x, area y, ... into the DURDzzzO data set.

(x-y)

Writes the segment data records for output area x, area x+1, ..., area y into the DURDzzzO data set.

(*)

Writes the segment data records for all output areas that are not specified on other FILECTL= control statements into the DURDzzzO data set.

Area numbers x and y are decimal numbers in the range of 1 - 2048.

Instead of DURDzzzO, you can specify XDzzzzzO.

Instead of DURDzzzE, you can specify XDzzzzzE if it is specified.

Instead of DURSzzzO, you can specify XSzzzzzO.

Empty area considerations

When there are no segment data records for output areas specified in FILECTL statement, the areas are regarded as empty. They are initialized during the reload, unless EMPTY=NO is specified on the USERCTL control statement. This consideration is applicable when the area is specified in FILECTL as in the following:

- 'ALL' is selected in the FILECTL statement.
- '*' is selected in the FILECTL statement.
- All areas are specified clearly in the FILECTL statement.

Default values

If FILECTL statements are not specified, each output area (number zzzzz) corresponds to its own DURDzzzO or XDzzzzzO data set. The DURDzzzO or XDzzzzzO data set contains all segments that FABCUR3 loads into area zzzzz, where zzzzz is the area number (field DMACRAID in the DMAC control block) assigned to that area during the processing of ACBGEN.

Error conditions

Duplicate references to an area or file in the FILECTL control statements are flagged with an error message, and cause program termination.

EXITRTN control statement

The optional EXITRTN statement specifies the name of the user exit routine that will be invoked with a COMPRESS function.

```
[EXITRTN=exit-routine]
```

EXITRTN=

This optional keyword specifies the name of the user exit routine to be invoked with a COMPRESS function.

exit-routine

Identifies the name of the user exit routine that will be called.

An EXITLIB DD statement must be provided when this control statement is specified.

For more information, see the topic "Exit routine option and its interface" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

IMSCOMP control statement

The optional IMSCOMP statement specifies whether the segment edit/compression routine will be invoked with segment compression call (entry code 0) for candidate Segments.

```
[IMSCOMP={NO|CMP}]
```

IMSCOMP=

This optional keyword determines whether the segment edit/compression routine will be invoked with segment compression call (entry code 0) for candidate segment.

NO

Specifies that the segment edit/compression routine is not invoked. IMSCOMP=NO is the default value.

CMP

Specifies that the segment edit/compression routine is invoked with segment compression call (entry code 0) for candidate segments.

USERCTL control statement

The optional USERCTL statement controls SDEP segment sequencing and whether FABCUR3 is to initialize empty areas.

```
USERCTL  
[SDEPSEQ={NORMAL|USER}]  
[EMPTY={NO|YES}]
```

USERCTL

This control statement has the following keywords.

SDEPSEQ=

This optional keyword controls the SDEP sequencing.

NORMAL

Specifies that the first SDEP segment inserted by FABCUR6 will be the first SDEP segment retrieved by IMS using GN processing. SDEPSEQ=NORMAL is the default value.

USER

Specifies that the first SDEP segment inserted by FABCUR6 will be the last SDEP segment retrieved by IMS using GN processing.

EMPTY=

This optional keyword determines whether area information records will be generated for areas that have no segment that causes FABCUR3 to initialize empty areas.

NO

Specifies that no area information records will be generated for areas that have no segment, and FABCUR3 will not initialize empty areas. EMPTY=NO is the default value.

YES

Specifies that area information records will be generated for areas that have no segment, and FABCUR3 will initialize empty areas.

FORMAT control statement

The optional FORMAT statement specifies the type of reloaded segment record format.

```
[FORMAT={DBT|TFMT}]
```

FORMAT=

The control statement specifies the type of reloaded segment record format.

DBT

Specifies that the format of the reloaded segment records is the same as IMS DBT 2.x. FORMAT=DBT is the default value.

TFMT

Specifies that the format of the reloaded segment records is enhanced, which means that the prefix part of the record is generated based on the maximum number of segment levels defined in DBD.

AREA_INFORMATION_RECORD control statement

The optional AREA_INFORMATION_RECORD statement specifies whether the area information record will be generated for areas.

```
AREA_INFORMATION_RECORD={YES|NO}
```

AREA_INFORMATION_RECORD=

The optional AREA_INFORMATION_RECORD statement specifies whether the area information record will be generated for areas.

YES

Specifies that area information records will be generated for areas. For empty areas, the EMPTY= keyword parameter of the USERCTL control statement decides whether to generate area information records for the areas. AREA_INFORMATION_RECORD=YES is the default value.

NO

Specifies that no area information records will be generated for areas. The EMPTY=YES keyword parameter of the USERCTL control statement will be ignored except when AREA_INFORMATION_RECORD=NO, FORMAT=TFMT, and LRECL=SEGTFMT control statements are specified.

The segment records that are created by FABCUR6 with the AREA_INFORMATION_RECORD=NO option can be reloaded with the FPB DEDB Reload utility, but such records cannot be reloaded with the FPA Reload function.

An abbreviation AIR is used for AREA_INFORMATION_RECORD.

LRECL control statement

```
LRECL={BLOCK|SEGTFMT}
```

LRECL=

The LRECL statement is composed of:

BLOCK

LRECL of a reload segment data set generated by FABCUR6 is determined as BLKSIZE - 4. LRECL=BLOCK is the system default value.

SEGTFMT

LRECL is determined on the basis of the maximum length of segments and the maximum number of segment levels defined in DBD by specifying LRECL=SEGTFMT together with the FORMAT=TFMT control statement.

OUTDD control statement

The OUTDD statement specifies whether the DD statements for the output data sets are mandatory.

```
OUTDD={YES|NO}
```

OUTDD=

This optional statement specifies whether the DD statements for the output data sets are mandatory.

YES

Specifies that the DD statement for the output data sets, which is DURDzzzO, XDzzzzzO, DURSzzzO, XSzzzzzO, or UR6DBDFN, is required. OUTDD=YES is the default value.

NO

Specifies that the DD statement for the output data sets, which is DURDzzzO, XDzzzzzO, DURSzzzO, XSzzzzzO, or UR6DBDFN, is optional.

Output for the HD To DEDB Unload Data Set Conversion utility

The following topics describe the output produced by FABCUR8.

UR6PRINT DD data set

The UR6PRINT data set contains messages that are issued by programs FABCUR8 and FABCUR6. This data set contains 133-byte records.

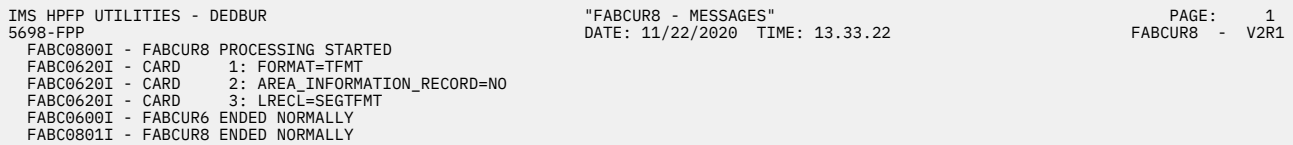
Format

If you code the block size in your JCL, it must be a multiple of 133. But, it is better to code your DD statement as follows:

```
//UR6PRINT DD SYSOUT=A
```

FABCUR8-Messages report

The following figure shows an example of the Messages report.



```
IMS HPFP UTILITIES - DEDBUR                                "FABCUR8 - MESSAGES"
5698-FPP                                                    DATE: 11/22/2020  TIME: 13.33.22
FABC0800I - FABCUR8 PROCESSING STARTED                      PAGE: 1
FABC0620I - CARD 1: FORMAT=TFMT                            FABCUR8 - V2R1
FABC0620I - CARD 2: AREA_INFORMATION_RECORD=NO
FABC0620I - CARD 3: LRECL=SEGTFMT
FABC0600I - FABCUR6 ENDED NORMALLY
FABC0801I - FABCUR8 ENDED NORMALLY
```

Figure 38. FABCUR8-Messages report

UR6AUDIT DD data set

HD To DEDB Unload Data Set Conversion utility generates a two-part Audit Control report to provide verification totals. This data set contains 133-byte records.

Format

If you code the block size in your JCL, it must be a multiple of 133. It is better to code your DD statement as follows:

```
//UR6AUDIT DD SYSOUT=A
```

FABCUR6 Audit Control report

The FABCUR6 Audit Control report has the following parts:

1. Part 1: SEGMENTS TO BE RELOADED TO DATABASE dbdname

This section of the report provides a count of the number of segments that are to be reloaded to each area of the new database and to the database total.

2. Part 2: SEGMENT TOTALS BY OUTPUT FILE

This section of the report provides segment counts and area totals by the output file ddname. File totals and a database total are also provided.

The area totals should match the area totals in Part 1. The file totals are ultimately verified against the reload file totals. The database total should match the total in Part 1.

If FILECTL statements are not used (that is, ddnames default to area numbers), there is only one area per file. Conversely, if FILECTL statements are used, a file may contain data for more than one area.

The following figure shows an example of the Audit Control report.

SEGMENTS TO BE RELOADED TO DATABASE DEDBJN22 :

AREA NO	AREA NAME	SEG CODE	SEG NAME	NUMBER OF SEGMENTS
1	DB22AR0	1	ROOTSEG1	2
		2	SDSEGNM1	0
		3	DD1	6
		4	DD2	6
		5	DD3	7
		6	DD4	4
		7	DD43	2
		8	DD44	0
		9	DD5	1
		10	DD53	0
** AREA TOTALS **				28
*** DATABASE TOTAL ***				28

Figure 39. FABCUR6 Audit Control report (Segments to be reloaded to database)

The following figure shows an example of the segment totals by output file section of the Audit Control report.

SEGMENT TOTALS BY OUTPUT FILE:

FILE DDNAME	AREA NO	SEG CODE	SEG NAME	NUMBER OF SEGMENTS
DURD0010	1	1	ROOTSEG1	2
		2	SDSEGNM1	0
		3	DD1	6
		4	DD2	6
		5	DD3	7
		6	DD4	4
		7	DD43	2
		8	DD44	0
		9	DD5	1
		10	DD53	0
** AREA TOTALS **				28
*** FILE TOTALS ***				28
**** DATABASE TOTAL ****				28

Figure 40. FABCUR6 Audit Control report (Segment totals by output file)

Example of the HD To DEDB Unload Data Set Conversion utility

The following figure shows example JCL statement for the HD To DEDB Unload Data Set Conversion utility.

The example JCL stream shown in the following figure converts an HD unload data set of the HD database HDDBJN22 to that of a DEDB database. All unloaded segment records will be written to the DURD0010 data set in a TFMT format.

```

//*****
//*   FABCUR8:                                     *
//*                                     *
//*   Convert HD unloaded file to DEDB USR file format   *
//*                                     *
//*****
//FABCUR8 EXEC PGM=FABCUR8,PARM='DEDBJN22,HDDBJN22'
//*-----*
//* For FABCUR8 DD statements *
//*-----*
//DURINPT DD DISP=SHR,DSN=HDDDB.HDDBJN22.UNLOAD
//DBDLIB DD DISP=SHR,DSN=IMSVS.DBDLIB
//ACBLIB DD DISP=SHR,DSN=IMSVS.ACBLIB
//*-----*
//* For FABCUR6 DD statements *
//*-----*
//RMODLIB DD DISP=SHR,DSN=IMSVS.SDFSRESL
//          DD DISP=SHR,DSN=IMSVS.PGMLIB
//DURD0010 DD DSN=HPFP.UR.FILE001.SEGDATA,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=TAPE
//UR6BDFN DD DSN=HPFP.UR.DURBDFN,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(1,1))
//DURS0010 DD DSN=HPFP.UR.FILE001.SORTCARD,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(1,1))
//UR6PRINT DD SYSOUT=*
//UR6AUDIT DD SYSOUT=*
//UR6CTL DD *
FILECTL=1,ALL
FORMAT=TFMT
/*
//

```

Figure 41. Sample JCL stream to convert an HD unload data set in an HD database to that of a DEDB database

Chapter 7. DEDB/HD Unload Conversion utility

Use the DEDB/HD Unload Conversion utility (FABCUR9) to load data from various formats of unload files onto an IMS full-function or Fast Path DEDB Database.

To load a Fast Path DEDB database, FABCUR9 (the DEDB/HD Unload Conversion) requires the Transaction Manager of the currently supported version of IMS.

Topics:

- [“Functions of the DEDB/HD Unload Conversion utility” on page 81](#)
- [“Data and system flow of the DEDB/HD Unload Conversion utility” on page 82](#)
- [“Restrictions of the DEDB/HD Unload Conversion utility” on page 82](#)
- [“Running the DEDB/HD Unload Conversion utility” on page 83](#)
- [“Considerations for creating databases of various formats” on page 83](#)
- [“PSB requirements” on page 85](#)
- [“DD statements for the DEDB/HD Unload Conversion utility” on page 85](#)
- [“Input for the DEDB/HD Unload Conversion utility” on page 86](#)
- [“Output for the DEDB/HD Unload Conversion utility” on page 95](#)
- [“Setting site default values for the DEDB/HD Unload Conversion utility” on page 97](#)
- [“Examples for the DEDB/HD Unload Conversion utility” on page 98](#)

Functions of the DEDB/HD Unload Conversion utility

The function of the DEDB/HD Unload Conversion utility (FABCUR9) is to load data from various formats of unload files onto an IMS full-function or Fast Path DEDB Database.

The layout of records in the unload file may be one of the following formats:

- FPB DEDB Unload/Reload with FORMAT=DBT
- FPB DEDB Unload/Reload with FORMAT=TFMT
- HD Reorganization Unload/Reload

This utility is designed to provide IMS users with an easy to use, flexible tool to load data onto a database whether the target database is online or offline.

It uses standard IMS DL/I calls to ensure that it will function correctly with all levels of IMS.

With this utility you can:

- Specify which segment types are to be inserted.
- Specify which segment types are to be replaced.
- Specify which segment types are to be bypassed.
- Use different segment names in the target database from those in the source database.
- Change the record format for segments in the target database from what was used in the source database.
- Run the utility as an online program (BMP) or as a batch program (DLI/DBB).
- Process data unloaded from DEDB Unload/Reload or the HD Reorganization Unload/Reload.
- Process data from any source which is converted to match formats created by any of DEDB Unload/Reload or the HD Reorganization Unload/Reload run.
- Use any PSB that has the appropriate segment sensitivity.
- Run in 'TEST' mode to preview the process, or to create the sequential flat file without performing any data base DL/I calls.

You can use the FABCUR9:

- To load a new HDAM, HIDAM, or HALDB database with data extracted from a DEDB database.
- To load a new HDAM, HIDAM, or HALDB database with data extracted from another HDAM or HIDAM full function database or HALDB.
- To load a new DEDB database with data extracted from another DEDB, HDAM, or HIDAM full function database or HALDB.
- To add records to an existing database.
- To load, insert, or update a subset of the segment types for a database.
- To update selected segment types en masse on a database.
- To populate a testing database.
- To create a flat file that contains database records.
- As an alternative to writing your own application to insert or update database records.

Data and system flow of the DEDB/HD Unload Conversion utility

This topic describes the data and system flow of the DEDB/HD Unload Conversion utility.

The following figure shows the FABCUR9 input, output, and processing flow.

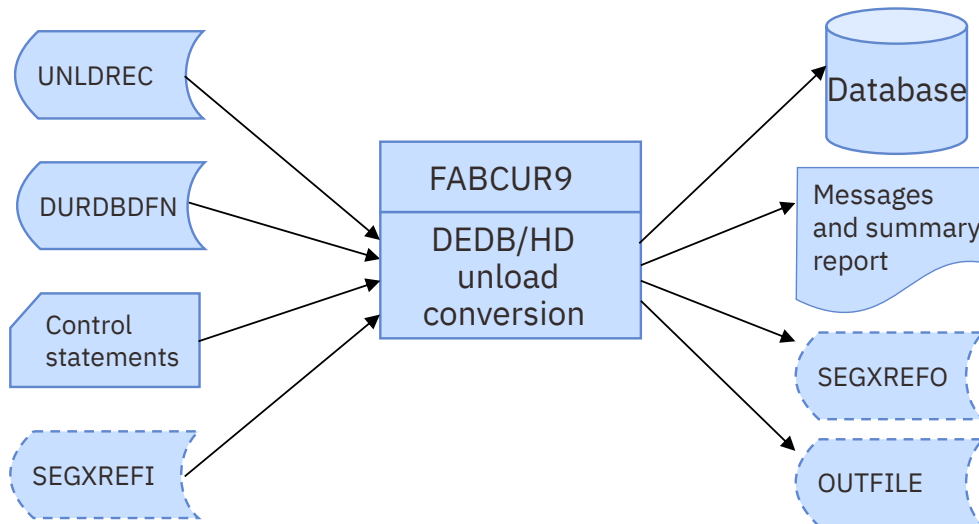


Figure 42. Flow of the DEDB/HD Unload Conversion utility

Restrictions of the DEDB/HD Unload Conversion utility

There are some restrictions when running the DEDB/HD Unload Conversion utility.

- Compressed unloaded segment records are not supported for DEDB, HDAM, HIDAM, or HALDB unloads.
- Segment Subset Pointers (SSP) are not reproduced, nor is the SSP information from an unloaded record loaded onto a target database.
- If only selected segment types are being loaded, all of the segment types in a given path must be selected; otherwise, the attempt may be made to load a child segment without loading the parent prior.
- Non-HD standard unload format is not supported for HD unloads.
- DEDB unloaded data sets containing segments having reached the Insert Limit Count (ILC) are not supported.
- Processing of SDEP physical records is not supported.

Running the DEDB/HD Unload Conversion utility

This topic describes how to run the DEDB/HD Unload Conversion utility.

About this task

To run the DEDB/HD Unload Conversion utility, the following steps are required:

- Invocation of the appropriate IMS supplied batch procedure, or equivalent:

BMP execution:

An EXEC statement of the IMS supplied procedure IMSBATCH

DLI execution:

An EXEC statement of the IMS supplied procedure DLIBATCH

DBB execution:

An EXEC statement of the IMS supplied procedure DBBBATCH

- Specification of the PSB which the appropriate sensitivity to the database and all of its segments.
- DD statements referencing the databases, if DLIBATCH or DBBBATCH procedures are being used.
- DD statements defining inputs and outputs

Then you can run the utility as follows:

Procedure

1. Code the JCL for the FABCUR9 job step.
2. Specify the DD statements to define input data sets, output data sets, and how the function is run.
3. Start the IMS online system if FABCUR9 is run as a BMP job.
4. Run the JCL.

Note: The DEDB/HD Unload Conversion utility (FABCUR9) supports the site default parameter. Macros and sample JCL streams are provided to generate the site default table.

Considerations for creating databases of various formats

This topic describes the considerations for creating various format databases by using the DEDB/HD Unload Conversion utility.

Creation of an HD database from a DEDB unload file

A PSB with a PROCOPT of 'L' or 'LS' is used to load a new HDAM or HIDAM database from unload records. A PSB with a PROCOPT of 'L' or 'LS' requires that the root records be added in the proper order. IMS may return a status code of 'LB' or 'LC' if it encounters root records in an improper order.

If the utility is being used to load a new HDAM or HIDAM database from a DEDB unload file, the records must be sorted in the sequence in which they are to be loaded.

This order may not be the same sequence in which the records were unloaded.

The following are considerations:

- If the target database is HDAM, make sure that the randomizer used by the DBD for the target is equivalent to the randomizer used by the unloaded database.
- If the target database is HIDAM, the unload file must be sorted into ascending root sequence.

If FPB DEDB Unload/Reload Utilities were used to create the unload file, a file containing sort control statements was created as a part of the unload process. These sort control statements can be modified to sort the unload in ascending root sequence.

To achieve this order, do the following:

1. Modify the sort control statements in the following manner:
 - a. Add 9 to 'position parameter' (first parameter in the FIELDS= statement). This position equates to the beginning of the concatenated key field in the unload record.
 - b. Subtract 9 from the 'length parameter' (second parameter in the FIELDS= statement)

For example, if the DEDB Unload/Reload utility generated the following sort control statements:

```
SORT FIELDS=(5,079,CH,A),SIZE=00000830
RECORD TYPE=V,LENGTH=(0482,,,0092)
```

they should be changed to the following:

```
SORT FIELDS=(14,070,CH,A),SIZE=00000830
RECORD TYPE=V,LENGTH=(0482,,,0092)
```
2. Precede the step that invokes FABCUR9 with a sort step

The SORTIN statement should reference the unload file.

The SORTOUT statement should reference a data set to be used as the target of the sort.

The SYSIN input should reference the modified sort control statements.
3. In the job step that runs FABCUR9, specify the SORTOUT data set on the UNLDREC DD statement.
 - If the target database is being updated (for example, REPL-type processing), the sequence of the root records in the file is not important; however, the dependent segments of these roots must be in correct sequence and hierarchical order.

Creation of a HIDAM database from an HDAM unload file

It cannot be assumed that the unload of an HDAM database is in ascending root key sequence. Consequently, it may not be possible to load the HIDAM database with a PSB with PROCOPT=L or PROCOPT=LS.

However, the load of a new HIDAM database from an HDAM unload can be accomplished through the following steps:

1. Use the DFSDDLTO program to load (PSB with PROCOPT=L or PROCOPT=LS) a dummy record.
2. Use the DFSDDLTO program to delete (PSB with PROCOPT=D or equivalent) the dummy record you just added.

The database is now primed, and you can add records using the IMS ISRT call.

3. Use FABCUR9 to ISRT (PSB with PROCOPT=I or equivalent) records into the database with the HDAM unload file.

Since you will not be using a PSB with PROCOPT=L or PROCOPT=LS, it is no longer necessary that the root keys be in ascending sequence.

Creation of a DEDB database from an HD unload file

A PSB with PROCOPT=I or equivalent is used to load a new DEDB database from HD or DEDB unload records.

The utility must be run as a BMP in order to process a DEDB database.

Make sure that the randomizer used by the DBD for the target is equivalent to the randomizer used by the unloaded database.

Recovery and restart

While CHKP is performed by the FABCUR9, the program cannot be restarted via the DL/I XRST call.

If the utility is being used to load records onto a new database, the database needs to be reallocated prior to attempting to rerun the utility.

However, in all other situations, the utility can simply be restarted from the beginning.
The FABCUR9 can be stopped by using the /STOP REGION command in BMP execution.

PSB requirements

This topic describes the PSB requirements for FABCUR9.

The following requirements apply to the PSB being used by the FABCUR9.

1. The language specified in the PSB by the LANG= parameter can be ASSEMB, COBOL, PL/I, or PASCAL.
2. If the I/O PCB is present, the utility will take checkpoints; otherwise it will not.
 - An I/O PCB is always present in BMP execution.
 - The I/O PCB will be present in DLI/DBB execution if CMPAT=YES is coded.
 - The I/O PCB will not be present in DLI/DBB execution if CMPAT=NO (the default) is taken.
3. The target DB PCB for the database being loaded can be anywhere in the PSB; however, the first occurrence of the DB PCB for this database will be used.
4. PROCOPT requirements:
 - For an initial load of an empty HDAM or HIDAM database, the PSB must be coded with PROCOPT=L or PROCOPT=LS for all of the target segments.
 - For an initial load of an empty DEDB, the PSB must be coded with PROCOPT=A for all of the target segments.

DD statements for the DEDB/HD Unload Conversion utility

DD statements for the DEDB/HD Unload Conversion utility (FABCUR9) determine the input and output data sets and how the utility is run.

The following table summarizes the DD statements for FABCUR9.

Table 14. FABCUR9 DD statements

DDNAME	Use	Format	Required or optional
JOBLIB/STEPLIB	Input	PDS	Required
UNLDREC	Input		Required
CNTLCRDS	Input	LRECL=80	Required
DURDBDFN	Input		Required
SEGXREFI	Input	LRECL=80	Optional
SYSPRINT	Output	LRECL=121	Required
SEGXREFO	Output	LRECL=80	Optional
OUTFILE	Output	RECV=VB,BLKSIZE=32760	Optional

JOBLIB/STEPLIB DD

Defines the library that contains the IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0).

UNLDREC DD

Defines the input data set that contains the unload records.

The records may originate from any of the following:

- The FPA Unload function and the FPB Unload utility with FORMAT=DBT
- The FPA Unload function and the FPB Unload utility with FORMAT=TFMT
- HD Reorganization Unload/Reload

- Any other source that can produce records in the format of one of the preceding unload/reload tools.

For the format of the data set records, see the documentation for these products.

CNTLCRDS DD

Defines the input data set that contains the CNTLCRDS control statements.

The statements are optional; the DD statement is required.

For CNTLCRDS control statements, see [“Control statements” on page 87](#).

DURDBDFN DD

Defines the input data set that contains the database definition record associated with the data base. The record is created by one of the FPA Unload function, the FPB Unload utility, the Database Definition Record Create utility, and the DEDB Reload Segment Data Set Create utility.

The DD statement is for processing a DEDB unload. It can be omitted if the Segment Cross-Reference data set is being used.

This DD statement may be omitted for processing an HD unload.

SEGXREFI DD

Defines the input data set that contains cross-reference of segment names with segment code identifiers. It may also contain Segment Cross-Reference keywords.

This data set is required only if the CNTLCRDS control statement 'SEGXREFI' has been coded.

SEGXREFO DD

Defines the output data set that contains a cross-reference of segment names with segment code identifiers.

This data set is required only if the CNTLCRDS control statement 'SEGXREFO' has been coded.

SYSPRINT DD

Defines the output data set that contains the Diagnostic Messages and Summary report.

OUTFILE DD

Defines the output data set that contains images of the segments that have been read from the unload file.

This data set is required only if the CNTLCRDS control statement 'OUTFILE' has been coded.

For more information about the OUTFILE records layout, see the topic "OUTFILE records layout" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

Related reference

[CNTLCRDS DD data set](#)

The CNTLCRDS data set contains the user's description of the database conversion processing to be done by FABCUR9.

Related information

[Segment Cross-Reference records](#)

This topic describes the optional Segment Cross-Reference records.

Input for the DEDB/HD Unload Conversion utility

You must specify the necessary input DD data sets to run FABCUR9.

CNTLCRDS DD data set

The CNTLCRDS data set contains the user's description of the database conversion processing to be done by FABCUR9.

Note: The default values for FABCUR9 control statement can be changed by using the site default table. For more information, see the topic "Site default support for FPB" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

Format

This control statement data set usually resides in the input stream. However, it can also be defined as a sequential data set or as a member of a partitioned data set. It must contain 80-byte, fixed-length records. Block size, if coded, must be a multiple of 80. This data set can contain several different types of control statement, including a comment statement. It is coded as shown in the following figure.

```
//CNTLCRDS DD *  
INPUT=FP  
DBDNAME=DATABAS1  
OUTFILE  
/*
```

Figure 43. FABCUR9 CNTLCRDS data set

Control statements

The FABCUR9 control statements are as follows:

- For indicating a data value
 - INPUT
 - DBDNAME
 - CHKP
- For indicating global actions
 - GHU
 - OUTFILE
 - REPL
 - SEGXREFI
 - SEGXREFO
 - TEST

FABCUR9 processing is determined by control statements, although in many cases, control statements may not be necessary. The control statements are read in from the input source specified in the CNTLCRDS DD statement.

All control statements are optional. If no control statements are supplied, FABCUR9 defaults to performing DL/I ISRT Calls to a database, and assumes the following:

- The unload file is made up of a Fast Path database and the format produced by DEDB Unload/Reload (INPUT=FP).
- The DBDNAME for the database is extracted from the DURDBDFN data set.

The following rules apply to control statements:

- Control statement records have a fixed length of 80 bytes.
- One or more control statements can be specified. A control statement record can include only one control statement.
- If specifying two or more keywords, separate them with one of the following:
 - a comma
 - one or more blanks
 - a comma and one or more blanks
- Comment statements can be marked with an asterisk (*) in column 1.
- Blank records are ignored.
- Control statement keywords are free form and are coded within the boundaries of columns 1 through 80, subject to the following syntactical rules:

- Keywords may start in any column.
- Keywords may occur in any order.
- There can be no intervening blank between a keyword indicating a data value and the value itself.
- Keywords and their associated values cannot span or continue on multiple control statement records.
- Unless specified, none of the control statements and/or keywords are case-sensitive.

The image of each record containing control statements will be written to the data set identified by the SYSPRINT DD statement.

Control statements for data value

Data value keywords influence how the run is processed.

INPUT control statement

This optional INPUT statement specifies the format of the unload records to be processed.

```
[INPUT={FP|FF|TFMT}]
```

INPUT=

This optional keyword is one of the following:

FP

The FPA Unload function and the FPB Unload utility with FORMAT=DBT. This is the default value.

FF

HD Reorganization Unload/Reload

TFMT

The FPA Unload function and the FPB Unload utility with FORMAT=TFMT keyword

DBDNAME control statement

This optional DBDNAME statement specifies the database to be processed.

```
[DBDNAME=dbdname]
```

DBDNAME=

This optional keyword specifies the DBD name of a database.

The DBDNAME keyword may be optional. For a Fast Path database, it defaults to the dbdname contained in the DURDBDFN. However, if a DURDBDFN is not used, or the unload file is from a full-function database, the DBDNAME= keyword is required.

CHKP control statement

This optional CHKP statement specifies the interval between root segment updates at which checkpoints are taken.

```
[CHKP={1000|nnnnnnn}]
```

CHKP=

This optional keyword specifies the interval between root segment updates at which checkpoints are taken. The nnnnnnn is a 1- to 7-digit number. The default is 1000.

Control statements for global action

Control statements for global action specify additional functions to be performed other than inserting unload records onto the database.

GHU

This optional keyword specifies that a GHU call should be issued for all segments in anticipation of a REPL call for all segments where the 'II' status code was returned from a DL/I ISRT call.

If the GHU keyword is omitted, the default is that no such action is taken. However, if the REPL keyword is used, the GHU keyword is implied, and the specified action will be taken even though the GHU keyword is not explicitly specified.

It is recommended that you code GHU wherever duplicate records are expected on the database. GHU will ensure that positioning in the database is retained.

OUTFILE

This optional keyword specifies that images of the segment data should be written to a sequential data set identified by the OUTFILE DD statement in the JCL.

REPL

This optional keyword specifies that a REPL call should be issued for all segments where the 'II' status code was returned from a DL/I ISRT call.

If the REPL keyword is omitted, the default is that no such action is taken. When REPL is specified, the DL/I REPL call will be preceded by the appropriate GHU DL/I call.

SEGXREFI

This optional keyword specifies that segment code or segment name data is being supplied in a sequential data set identified by the SEGXREFI DD statement in the JCL.

SEGXREFO

This optional keyword specifies that segment code or segment name data is to be written to a sequential data set identified by the SEGXREFO DD statement in the JCL.

TEST

This optional keyword specifies that no DL/I database update calls are to be performed, but that all other functions are to continue processing.

SEGREFI DD data set

The SEGREFI data set contains the user's description of the special processing that FABCUR9 is to do for the segments.

Segment Cross-Reference records

This topic describes the optional Segment Cross-Reference records.

Before attempting to use these functions, you should read the preceding topics so that you understand the purpose of each Unload/DB Load functions and how it relates to other functions.

Situations in which a Segment Cross-Reference File is required

A Segment Cross-Reference File is required in the following circumstances:

- When segment names in the target database are different from the comparable segments in the unloaded database
- When selected segments are not to be processed
- When not all segments are to be processed in the same manner
- When the segment format in the target database is different from the comparable segment in the unloaded database

Here are some conditions and characteristics about the Segment Cross-Reference records:

- All segments defined in the DURDBDFN data set must be described in the Segment Cross-Reference records.
- If the Segment Cross-Reference Records are not used, comparable data is extracted from the database definition record (DURDBDFN) for DEDB unloads, or from a combination of the header and segment records for an HD unload. However, DURDBDFN records may be IMS level dependent. If the necessary data from the DURDBDFN record is captured via using the SEGXREFO CNTLCRDS control keyword,

processing can be driven during subsequent executions of DEDB Unload/DB Load by the Segment Cross-Reference Records.

- In addition, some of the function afforded by the Segment Cross Reference Keywords is only available if the Segment Cross-Reference Records are used.
- If the CNTLCRDS keyword SEGXREFI is encountered, Segment Cross-Reference Records are read from the SEGXREFI data set.
- If the CNTLCRDS keyword SEGXREFO is encountered, Segment Cross-Reference Records are written to the SEGXREFO data set.
- If segment format (variable length or fixed length) is different between the source database segment and the target database segment, then you must specify the SEGXREFI option and you must provide the segment cross reference records.
- Segment Cross-Reference Records are 80-byte fixed-length.
- Comment statements may be specified with an asterisk (*) in column 1.
- Blank records are not allowed.
- There is a fixed format and variable format area of these records.
- The Fixed Format area contains column-specific variables.
- The Variable Format Area may contain Segment Cross-Reference keywords.

Example of a Segment Cross-Reference table

The following figure shows an example of a Segment Cross-Reference table.

```
001 001 000 SEGMENT1 V F 00044 00000 00000
002 002 001 SEGMENT2 V S 00044 00000 00000
003 002 001 SEGMENT3 V 00044 00000 00000 GHU
004 002 001 SEGMENT4 V 00044 00000 00000 REPL
005 002 001 SEGMENT5 V 00044 00000 00000
006 002 001 SEGMENT6 V 00044 00000 00000
007 002 001 SEGMENT7 V 00044 00000 00000 BYPASS
008 002 001 SEGMENT8 V 00044 00000 00000
```

Figure 44. Example of a Segment Cross-Reference table

The format of the Segment Cross-Reference Table is as follows:

- Fixed Format Area

Position

Definition

1-3

Segment code

4

Not used

5-7

Level of this segment in the hierarchy

8

Not used

9-11

Segment code of the parent segment

12

Not used

13-20

Segment name

21

Not used

- 22**
Source record format of the segment
- 23**
Not used
- 24**
Target record format of the segment
- 25**
Not used
- 26**
Type of the segment
- 27**
Not used
- 28-32**
Starting position of the segment in the unload file
- 33**
Not used
- 34-38**
Minimum length of the segment
- 39**
Not used
- 40-44**
Maximum length of the segment
- 45**
Not used
- Variable Format Area
 - Position**
 - Definition**
 - 46**
Beginning of the area in which Segment Cross-Reference keywords can be coded

Fixed format area

There are multiple fields in the Fixed Format Area:

1. Seg Code Id

This is a 3-character numeric field which corresponds to the segment code assigned to a segment during DBD generation.

If Segment Cross-Reference Records are being generated by use of the SEGXREFO CNTLCRDS keyword, this field will be initialized with the seg code ID from one of the following sources:

- The DURDBDFN data set, if INPUT=FP or INPUT=TFMT is specified
- The HD Reorganization Unload/Reload unload file, if INPUT=FF is specified

Restrictions:

- This field must begin in column 1. It need not be padded with zeros; however, it must be right-aligned.
- This field must contain only numeric values.

2. Level in Hierarchy

This is a 3 character numeric field which corresponds to the level in the hierarchy for this segment.

If Segment Cross-Reference Records are being generated via use of the SEGXREFO CNTLCRDS keyword, this field will be initialized with the hierarchy level from one of the following sources:

- The DURDBDFN data set, if INPUT=FP or INPUT=TFMT is specified
- The HD Reorganization Unload/Reload unload file, if INPUT=FF is specified

Restrictions:

- This field must begin in column 5. It need not be padded with zeros; however, it must be right-aligned.
- This field must contain only numeric values.

3. Parent Seg Code Id

This is a 3 character numeric field which corresponds to the segment code assigned to the parent of this segment during DBD generation.

If Segment Cross-Reference Records are being generated via use of the SEGXREFO CNTLCRDS keyword, this field will be initialized with the hierarchy level from one of the following sources:

- The DURDBDFN data set, if INPUT=FP or INPUT=TFMT is specified
- The HD Reorganization Unload/Reload unload file, if INPUT=FF is specified

Restrictions:

- This field must begin in column 9. It need not be padded with zeros; however, it must be right-aligned.
- This field must contain only numeric values.

4. Seg Name

This is an 8-character name that corresponds to the segment name contained in the DBD.

If Segment Cross-Reference Records are being generated via use of the SEGXREFO CNTLCRDS keyword, this field will be initialized with the seg name from one of the following sources:

- The DURDBDFN data set, if INPUT=FP or INPUT=TFMT is specified
- The HD Reorganization Unload/Reload unload file, if INPUT=FF is used

This Seg name is used in the DL/I call to the target database when the Segment Cross-reference file is being used as input. When the Segment Cross-Reference records are used as input, the Seg names must match the names of the target segments in the database being loaded. Once the Segment Cross-Reference records have been created, the Seg name associated with each record can be changed from the name with which it was generated (source segment) to the name of the segment in the target database.

Restriction:

This field must begin in column 13.

5. Source Record Format

This is a single character that describes the record format of the database segments as they are in the unload file. The character must be:

F

If the segment is fixed length

V

If the segment is variable length

If Segment Cross-Reference Records are being generated via use of the SEGXREFO CNTLCRDS keyword, this field will be initialized to the following:

- If INPUT=FP or INPUT=TFMT is specified, the format will be determined by the contents of the DURDBDFN record.
- If INPUT=FF is specified, the source record format will be determined from the unload records themselves.

Restriction:

This field must be in column 22.

6. Target Record Format

This is a single character that describes the record format of the database segments as they are to be loaded in the target database. The character must be:

F

If the segment is fixed length

V

If the segment is variable length

blank

If you want to use the same value as the source record format

If Segment Cross-Reference Records are being generated via use of the SEGXREFO this field will be initialized to the corresponding source record format.

The Record Format field is used to determine whether fixed or variable length segments are loaded to the target database.

At execution time, if this field is blank, the source record format specification will be used for the target record format. An incorrect specification will result in message FABC0919W.

Restriction:

This field must be in column 24.

7. Record Type

This is a single character name that identifies the type of segment as one of the following:

S

If the segment is an SDEP segment

U

If the segment is an unkeyed segment

blank

If the segment is not an SDEP segment or an unkeyed segment

The record type may restrict certain types of processing in FABCUR9:

- FABCUR9 does not support the load of SDEP physical segments.
- FABCUR9 does not support update-type processing on unkeyed segments.

REPL or GHU type processing requested for an unkeyed segment will result in message FABC0929E or FABC0930E. Processing will terminate with condition code 8.

Restrictions:

- This field must be in column 26.
- The information related to whether a segment is keyed or not is not available in an HD unload. Consequently, during SEGXREFO processing of an HD unload file, the value for this field cannot be determined. In this case, the field is set to hex zeros ('00'X). The field must be changed to the appropriate value before the segment cross-reference file created in this manner can be used as subsequent input.

8. Data Starting Position

This is a 5-character numeric which identifies the position in the unload record at which the segment data begins.

If Segment Cross-Reference Records are being generated via use of the SEGXREFO CNTLCRDS keyword, this field will be initialized to the location of the data in the unload records.

Restrictions:

- This field must be in column 28. It need not be padded with zeros; however, it must be right-aligned.

- This field must contain only numeric values.

9. Minimum Length

This is a 5 character numeric which describes the minimum length of the data associated with the segment.

Restrictions:

- This field must be in column 34. It need not be padded with zeros; however, it must be right-aligned.
- This field must contain only numeric values; however, the content is not used.

10. Maximum Length

This is a 5 character numeric which describes the minimum length of the data associated with the segment.

Restrictions:

- This field must be in column 40. It need not be padded with zeros; however, it must be right-aligned.
- This field must contain only numeric values; however, the content is not used.

Variable Format Area—Segment Cross-Reference keywords

The Variable Format Area of the Segment Cross-Reference records may contain Segment Cross-Reference keywords.

Segment Cross-Reference keywords influence additional functions to be performed other than inserting unload records onto the database.

Segment Cross-Reference keywords are optional. If they are omitted, the optional action will not be performed.

Segment Cross-Reference keywords are free form and are coded within the boundaries of columns 46 through 80, subject to the following syntactical rules:

1. Keywords may start in any column.
2. There can be no intervening blanks between keywords indicating a data value and the value itself.
3. Two or more keywords are separated by either:
 - a comma
 - one or more blanks
 - a comma and one or more blanks
4. Keywords and their associated values cannot span or continue on multiple Segment Cross-Reference Records.

Unless specified, none of the Segment Cross-Reference Keywords and/or keywords are case sensitive.

The Segment Cross-Reference keywords are the following:

GHU

This keyword specifies that a GHU call should be issued for this segment in anticipation of a REPL call for all segments where the 'II' status code was returned from an DL/I ISRT call.

The GHU keyword is optional. If omitted, it defaults to no such action. However, if the REPL keyword is used, the GHU keyword is implied, and the specified action will be performed even though the GHU keyword is not explicitly specified.

It is recommended to code GHU in those instances where duplicate records are expected on the database. GHU will ensure that positioning in the database is retained.

Restriction:

None.

BYPASS

This keyword specifies that no attempt should be made to update this segment.

It is very useful if the PSB being used for a database update does not have sensitivity to certain segments. The BYPASS keyword will ensure that no attempt is made to access these segments, assuming that these segments are not parents of other segments for which processing is desired.

The BYPASS keyword is optional. If omitted, it defaults to no such action.

Restriction:

The BYPASS keyword should not be specified for a segment if the processing of segments dependent upon this segment is required. Correct positioning in the database may not be achieved if BYPASS is coded for segments in the hierarchical chain.

NOREP

This keyword specifies that no attempt should be made to replace this segment.

The NOREP keyword is optional. It is used to override a REPL setting should this setting be propagated to this segment via the presence of the REPL CNTLCRDS Global Action keyword.

Unlike with the BYPASS keyword, DL/I calls are issued for segments with the NOREP keyword; however, REPL processing will not occur for such segments.

Restriction:

None.

REPL

This keyword specifies that a REPL call should be issued for this segment for all segments where the 'II' status code was returned from an DL/I ISRT call.

The REPL keyword is optional. If omitted, it defaults to no such action. When REPL is specified, the DL/I REPL call will be preceded by the appropriate GHU DL/I call.

Restriction:

None.

Output for the DEDB/HD Unload Conversion utility

The following topic describes the output produced by FABCUR9.

SYSPRINT DD data set

This data set contains the Diagnostic Messages and Summary report created by the FABCUR9.

Format

This data set contains 121-byte records. If you code the block size in your JCL, it must be a multiple of 121. It is better to code your DD statement as follows:

```
//SYSPRINT DD SYSOUT=A
```

FABCUR9 Diagnostic Messages and Summary report

The following figure is an example of the report:

```

IMS HPFP UTILITIES - DEDBUR                "DEDB/HD UNLOAD DIAGNOSTIC MESSAGES AND SUMMARY REPORT"        PAGE:      1
5698-FPP                                DATE: 11/22/2020   TIME: 03.59.43        FABCUR9 -  V2R1

FABC0900I CNTLCRD:  CHKP=500  OUTFILE
FABC0900I CNTLCRD:  INPUT=FP  SEGXREFO
FABC0900I CNTLCRD:  REPL

FORMAT OF UNLDREC: FP

NUMBER OF SEGMENT TYPES:      8
  DBDNAME: DATABASE1
    DBDNAME SUPPLIED FROM DURDBDFN
    SEGMENT NAMES SUPPLIED FROM DURDBDFN
NUMBER OF LEVELS IN DATABASE HIERARCHY:      2
NUMBER OF SEGXREFO RECORDS GENERATED:      8

NAME      CODE    LVL  PAR  READ CNT  ISRT CNT  REPL CNT  SELECTED OPTIONS
SEGMENT1   1      1    0    1440      1440      0    REP-DUP
SEGMENT2   2      2    1      0          0      0    REP-DUP
SEGMENT3   3      2    1   1082     1082      0    REP-DUP
SEGMENT4   4      2    1      0          0      0    REP-DUP
SEGMENT5   5      2    1      0          0      0    REP-DUP
SEGMENT6   6      2    1      0          0      0    REP-DUP
SEGMENT7   7      2    1   2849     2849      0    REP-DUP
SEGMENT8   8      2    1     83         83      0    REP-DUP
TOTAL:      0      0    0    5454     5454      0
                END DFSRRC00/FABCUR9

```

Figure 45. FABCUR9 Diagnostic Messages and Summary report

In this example:

- FABC0900I messages display images of the CNTLCRDS control records.

In this example:

- The checkpoint frequency was overridden to 500.
- A sequential data set copy of the records was written to OUTFILE.
- Copies of the Segment Cross-Reference records were written to SEGXREFO.
- The unload file was specified as FP (DEDB Unload/Reload).
- Globally, any duplicate record encountered should be replaced.
- The DBDNAME of the database is DATABASE1.
- The DBDNAME came from the DURDBDFN data set.

Alternatively the DBDNAME could have come from the DBDNAME= CNTLCRDS control keyword.

- The Segment names came from the DURDBDFN data set.

Alternatively, the Segment names could have come from:

- The SEGXREFI data set (if CNTLCRDS control keyword 'SEGXREFI' was used)
- The Unload file (in the case of FPB DEDB Unload/Reload)
- The IMS LEVEL of 610 was extracted from the DURDBDFN.
- The Unload record format was FP.

The following other formats are possible:

- FF
- TFMT
- The number of types of segments in the DBD was 8.
- There were 2 levels in the database hierarchy.
- Since CNTLCRDS keyword SEGXREFO was included, 8 Segment Cross-Reference segments were generated.
- The statistics of each segment are reported.
 - The segment names are SEGMENT1 - SEGMENT8, with segment codes of 1 - 8.
 - The level in the hierarchy, and the segment code of the parent for each segment is given.
 - The column "SELECTED" shows the Segment Cross-Reference Keywords or Global Action Keywords that are active for this segment.

The following values may be presented:

- BYPASS

The BYPASS keyword was active for this segment.

- GHU-DUP

The GHU keyword was active for this segment.

- REP-DUP

The REPL keyword was active for this segment.

- NO-REP

The NOREPL keyword was active for this segment.

- TEST

The TEST keyword was active for this segment.

- SDEP

This is an SDEP segment.

Setting site default values for the DEDB/HD Unload Conversion utility

The DEDB/HD Unload Conversion utility allows you to specify site default values. Macros and sample JCL streams are provided to generate the site default table.

If you want to change the default values for control statements, use macro FABCOP9M and sample JCL FABCOP9J and generate a site default table.

The generated site default table library must be concatenated to the IMS HP Fast Path Utilities load module library in the JOBLIB or STEPLIB DD statement.

Use the TABLESET= parameter to specify the type of the table to generate. The keywords for the TABLESET= parameter are as follows:

USER

Builds a site default table. This is the default value.

SYSTEM

Builds a system default table that is to be used internally by the FABCUR9 program. Users of FABCUR9 should not specify this value.

DSECT

Builds a DSECT to map default table entries. Users of FABCUR9 should not specify this value.

When coding the macros, note the following:

- Under TABLESET=USER, specifying system default value will cause FABD3675I message to be generated and a table entry for the keyword value will not be generated.
- Under TABLESET=USER, coding the same macro more than once will cause FABD3676E message to be generated and will end with return code of 8. All necessary site default values for a macro must be specified in the same macro.

FABCOP9M macro

The following control statements can be specified:

INPUT= or FORMAT=

Specifies the format of the unloaded segment records.

FP

FPB DEDB Unload/Reload with FORMAT=DBT. This is the system default value.

FF

HD Reorganization Unload/Reload.

TFMT

FPB DEDB Unload/Reload with FORMAT=TFMT.

CHKP

Specifies the interval between root segment updates at which checkpoints are taken.

NNNNNNN

A 1- to 7-digit number. The system default value is 1000.

Examples for the DEDB/HD Unload Conversion utility

This topic shows examples for the DEDB/HD Unload Conversion utility.

The following figure shows example JCL for FABCUR9. The statements preceded with *opt might be needed based on the control statements included in the input stream.

The following figure shows example JCL stream for FABCUR9 running as a BMP.

```
//FABCUR9 EXEC IMSBATCH,
//          MBR=FABCUR9,PSB=LOADPSB      <-- IDENTIFY PSB
//STEPLIB DD DISP=SHR,DSN=dsname          <-- Unload/Reload Utilities
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*                    <-- Messages
//UNLDREC DD DISP=SHR,DSN=unldrec          <-- Unload record input
//CNTLCRDS DD *                           <-- Control stmts (input)
//DURDBDFN DD DISP=SHR,DSN=DURDBDFN        <-- DURDBDFN
*opt //SEGXREFI DD *                       <-- Seg cross ref (input)
*opt //SEGXREF0 DD DISP=SHR,DSN=segxrefo    <-- Seg cross ref (output)
*opt //OUTFILE DD DISP=SHR,DSN=outfile      <-- Seq data set (output)
```

Figure 46. Example JCL for FABCUR9 running as a BMP

The following figure shows example JCL for FABCUR9 running in DLI batch.

```
//FABCUR9 EXEC DLIBATCH,
//          MBR=FABCUR9,PSB=LOADPSB      <-- IDENTIFY PSB
//STEPLIB DD DISP=SHR,DSN=dsname          <-- Unload/Reload Utilities
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*                    <-- Messages
//UNLDREC DD DISP=SHR,DSN=unldrec          <-- Unload record input
//CNTLCRDS DD *                           <-- Control stmts (input)
//DURDBDFN DD DISP=SHR,DSN=DURDBDFN        <-- DURDBDFN
//ddname1 DD DISP=OLD,DSN=dbname1          <-- 1st dd of database
.
.
.
//ddnamex DD DISP=OLD,DSN=dbnamex          <-- last dd of database
*opt //SEGXREFI DD *                       <-- Seg cross ref (input)
*opt //SEGXREF0 DD DISP=SHR,DSN=segxrefo    <-- Seg cross ref (output)
*opt //OUTFILE DD DISP=SHR,DSN=outfile      <-- Seq data set (output)
```

Figure 47. Example JCL for FABCUR9 running in Batch (DLI)

The following topics describe the outcome of various combinations of keywords specified on your CNTLCRDS control statements.

In each example, there might be other parameters for which defaults are taken, but only selected defaults are listed.

For a complete discussion of the default values, see the information for each specific keyword in [“Control statements” on page 87](#).

Related reference

[CNTLCRDS DD data set](#)

The CNTLCRDS data set contains the user's description of the database conversion processing to be done by FABCUR9.

Example 1: Using the database definition record (DURDBDFN)

These are example JCL statements for using the database definition record (DURDBDFN).

The following examples use a DEDB unload file as input. In these examples, the database description is taken from the database definition record (DURDBDFN); the Segment Cross-Reference File (SEGXREFI) is not used.

Case 1

In this example, no control statement is specified.

```
<No statement was specified.>
```

Figure 48. Control statements: Database definition record (Case 1)

As a result, the following actions were taken:

- The format in which the unload was expected is FP (FPB DEDB Unload/Reload Utility format)
- All database format information was extracted from the database definition record.
- Only Load or ISRT processing occurred.
- No output was written to any sequential data sets.
- The Segment Cross-Reference table (SEGXREFI) was not read.

Case 2

In this example, control statements are specified as shown in the following figure.

```
TEST  
OUTFILE
```

Figure 49. Control statements: Database definition record (Case 2)

As a result, the following actions were taken:

- The format in which the unload was expected was FP (FPB DEDB Unload/Reload Utility format).
- All database format information was extracted from the database definition record.
- No Database DL/I calls were issued.
- Images of the segment data were written to the OUTFILE data set.
- The Segment Cross-Reference table (SEGXREFI) was not read.

Case 3

In this example, control statements are specified as shown in the following figure.

```
DBDNAME=DATABAS1  
OUTFILE  
SEGXREFO
```

Figure 50. Control statements: Database definition record (Case 3)

As a result, the following actions were taken:

- The format in which the unload was expected was FP (FPB DEDB Unload/Reload Utility format).
- The database name DATABAS1, which is specified in a control statement was used as the target database.

- All other database format information was extracted from the database definition record.
- Only Load or ISRT processing occurred.
- Images of the segment data were written to the OUTFILE data set.
- The Segment Cross-Reference table (SEGXREFI) was not read.
- The Segment Cross-Reference table was built based upon data extracted from the database definition record.

Case 4

In this example, control statements are specified as shown in the following figure.

```
INPUT=FP
OUTFILE
SEGXREF0
REPL
```

Figure 51. Control statements: Database definition record (Case 4)

As a result, the following actions were taken:

- The format in which the unload was expected was FP (FPB DEDB Unload/Reload format).
- All database format information was extracted from the database definition record.
- An IMS Status Code of 'II' returned on an ISRT DL/I call for any segment resulted in the program issuing a REPL DL/I call to replace.
- Images of the segment data were written to the OUTFILE data set.
- The Segment Cross-Reference table (SEGXREFI) was not read.
- The Segment Cross-Reference table was built based upon data extracted from the database definition record.

Example 2: Using an HD unload file

The following figure shows example JCL statement for using an HD unload file.

The following example uses an HD Reorganization Unload/Reload file as input. The database description is interpreted from the header record in the unload file. In this example, control statements are specified as shown in the following figure.

```
INPUT=FF
DBDNAME=DATABAS1
OUTFILE
SEGXREF0
REPL
```

Figure 52. Control statements: HD unload file

The report generated will be as shown in the following figure.

FABC0900I CNTLCRDS: INPUT=FF
FABC0900I CNTLCRDS: DBDNAME=DATABAS1
FABC0900I CNTLCRDS: OUTFILE
FABC0900I CNTLCRDS: SEGXREFO
FABC0900I CNTLCRDS: REPL

NUMBER OF SEGMENT TYPES: 8
DBDNAME: DATABAS1 IMS LEVEL: 610
DBDNAME SUPPLIED FROM CNTLCRDS
SEGMENT NAMES SUPPLIED FROM UNLDREC
FORMAT OF UNLDREC: FF

NUMBER OF SEGXREFO RECORDS GENERATED: 8
NUMBER OF LEVELS IN DATABASE HIERARCHY: 2

NAME	CODE	LVL	PAR	READ CNT	ISRT CNT	REPL CNT	SELECTED OPTIONS
SEGMENT1	1	1	0	1440	0	1440	REP-DUP
SEGMENT2	2	2	1	0	0	0	REP-DUP
SEGMENT3	3	2	1	1082	0	0	GHU-DUP
SEGMENT4	4	2	1	0	0	0	REP-DUP
SEGMENT5	5	2	1	0	0	0	REP-DUP
SEGMENT6	6	2	1	0	0	0	REP-DUP
SEGMENT7	7	2	1	2849	0	0	BYPASS
SEGMENT8	8	2	1	83	0	83	REP-DUP
TOTAL:	0	0	0	5454	0	1523	REP-DUP

Figure 53. FABCUR9 Diagnostic Messages and Summary report

As a result, the following significant actions were taken:

- The format in which the unload was expected is FF—HD Reorganization Unload/Reload format.
- All database format information was extracted from the database definition record.
- An IMS Status Code of 'II' returned on an ISRT DL/I call for any segment will result in the program issuing a REPL DL/I call to replace
- Images of the segment data were written to the OUTFILE data set.
- The Segment Cross Reference table (SEGXREFI) was not read.
- The Segment Cross Reference table was built from data extracted from the database definition record.

Example 3: Using the Segment Cross-Reference table

These are example control statements for using the Segment Cross-Reference table.

The examples in this topic assume that the Segment Cross-Reference table shown in the following figure is read for the database descriptions:

```
<--- Fixed format -----><--- Variable ---->
      area                      format area
Columns:
1   5   9   13   22   28   34   40   46
|   |   |   |   |   |   |   |   |
001 001 000 SEGMENT1 V   00044 00000 00000
002 002 001 SEGMENT2 V   00044 00000 00000 REPL
003 002 001 SEGMENT3 V   00044 00000 00000 GHU
004 002 001 SEGMENT4 V   00044 00000 00000
005 002 001 SEGMENT5 V   00044 00000 00000
006 002 001 SEGMENT6 V   00044 00000 00000
007 002 001 SEGMENT7 V   00044 00000 00000 BYPASS
008 002 001 SEGMENT8 V   00044 00000 00000
```

Figure 54. Segment Cross-Reference table as input (Cases 1-6)

In this example,

- Fixed Format Area
 - Columns 1-3 represent the segment code
 - Columns 5-7 represent the level of this segment in the hierarchy
 - Columns 9-11 represent the segment code of the parent segment
 - Columns 13-20 represent the segment name
 - Column 22 represents the source record format of the segment

- Columns 28-32 represent the starting position of the segment in the unload file
- Columns 34-38 represent the minimum length of the segment
- Columns 40-44 represent the maximum length of the segment
- Variable Format Area
 - Column 46 represents the beginning of the area in which Segment cross-reference keywords may be coded

The following significant actions are specified:

- SEGMENT2 is to be replaced when the DL/I returns an 'II' status code from an ISRT call.
- SEGMENT3 is to issue a GHU call to retain position in the database when DL/I returns an 'II' status code from an ISRT call.
- No DL/I calls are to be issued for SEGMENT 7.

Case 1

An empty database was the target. The control statements in the following example were specified:

```
DBDNAME=DATABAS1
OUTFILE
SEGXREFO
SEGXREFI
```

Figure 55. Control statements: Segment Cross-Reference table (Case 1)

The Diagnostic Messages and Summary report generated will be as in the following figure.

```
IMS HPFP UTILITIES - DEDBUR                                "DEDB/HD UNLOAD DIAGNOSTIC MESSAGES AND SUMMARY REPORT"    PAGE:      1
5698-FPP                                                    DATE: 11/22/2020  TIME: 03.59.43    FABCUR9 -  V2R1
```

```
FABC0900I CNTLCRDS: DBDNAME=DATABAS1
FABC0900I CNTLCRDS: OUTFILE
FABC0900I CNTLCRDS: SEGXREFO
FABC0900I CNTLCRDS: SEGXREFI
```

```
FABC0900I SEGXREFI: 001 001 000 SEGMENT1 V    00044 00000 00000
FABC0900I SEGXREFI: 002 002 001 SEGMENT2 V    00044 00000 00000 REPL
FABC0900I SEGXREFI: 003 002 001 SEGMENT3 V    00044 00000 00000 GHU
FABC0900I SEGXREFI: 004 002 001 SEGMENT4 V    00044 00000 00000
FABC0900I SEGXREFI: 005 002 001 SEGMENT5 V    00044 00000 00000
FABC0900I SEGXREFI: 006 002 001 SEGMENT6 V    00044 00000 00000
FABC0900I SEGXREFI: 007 002 001 SEGMENT7 V    00044 00000 00000 BYPASS
FABC0900I SEGXREFI: 008 002 001 SEGMENT8 V    00044 00000 00000
```

```
FORMAT OF UNLDREC: FP

NUMBER OF SEGMENT TYPES:      8
  DBDNAME: DATABAS1
  DBDNAME SUPPLIED FROM DURDBDFN
  SEGMENT NAMES SUPPLIED FROM SEGXREFI FILE
NUMBER OF SEGXREFO RECORDS GENERATED:      8
NUMBER OF LEVELS IN DATABASE HIERARCHY:      2
```

NAME	CODE	LVL	PAR	READ CNT	ISRT CNT	REPL CNT	SELECTED OPTIONS
SEGMENT1	1	1	0	1440	1440	0	
SEGMENT2	2	2	1	0	0	0	REP-DUP
SEGMENT3	3	2	1	1082	1082	0	GHU-DUP
SEGMENT4	4	2	1	0	0	0	
SEGMENT5	5	2	1	0	0	0	
SEGMENT6	6	2	1	0	0	0	
SEGMENT7	7	2	1	2849	0	0	BYPASS
SEGMENT8	8	2	1	83	83	0	
TOTAL:	0	0	0	5454	2605	0	

Figure 56. FABCUR9 Diagnostic Messages and Summary report (Case 1)

The following significant actions were taken:

- The format in which the unload was expected was FP—the FPB DEDB Unload/Reload Utility format
- The database name DATABAS1, which is specified in the CNTLCRDS was used as the target database.
- All other database format information was extracted from the Segment Cross-Reference Table.
- REPL processing was allowed for SEGMENT2; however, since the database was empty, only Load or ISRT processing was applicable.
- Images of the segment data were written to the OUTFILE data set.

- The Segment Cross-Reference table was rebuilt based on the data extracted from the SEGXREFI data set.

Case 2

The same CNTLCRDS were specified as in “Case 1” on page 102. This time, however, the target database had already been populated. No global actions were specified for the segments. The control statements in the following figure were specified.

```
DBDNAME=DATABAS1
OUTFILE
SEGXREFO
SEGXREFI
```

Figure 57. Control statements: Segment Cross-Reference table (Case 2)

The Diagnostic Messages and Summary report generated will be as in the following figure.

```
IMS HPFP UTILITIES - DEDBUR          "DEDB/HD UNLOAD DIAGNOSTIC MESSAGES AND SUMMARY REPORT"    PAGE: 1
5698-FPP                                DATE: 11/22/2020  TIME: 03.59.43    FABCUR9 - V2R1

FABC0900I CNTLCRDS:  DBDNAME=DATABAS1
FABC0900I CNTLCRDS:  OUTFILE
FABC0900I CNTLCRDS:  SEGXREFO
FABC0900I CNTLCRDS:  SEGXREFI

FABC0900I SEGXREFI: 001 001 000 SDWCACA V 00044 00000 00000
FABC0900I SEGXREFI: 002 002 001 SDWCAOA V 00044 00000 00000 REPL
FABC0900I SEGXREFI: 003 002 001 SDWCAAA V 00044 00000 00000 GHU
FABC0900I SEGXREFI: 004 002 001 SDWCACO V 00044 00000 00000
FABC0900I SEGXREFI: 005 002 001 SDWCACU V 00044 00000 00000
FABC0900I SEGXREFI: 006 002 001 SDWCALT V 00044 00000 00000
FABC0900I SEGXREFI: 007 002 001 SDWCALM V 00044 00000 00000 BYPASS
FABC0900I SEGXREFI: 008 002 001 SDWCAEM V 00044 00000 00000

FORMAT OF UNLDREC: FP

NUMBER OF SEGMENT TYPES:      8
DBDNAME: DDWCAD
DBDNAME SUPPLIED FROM DURDBDFN
SEGMENT NAMES SUPPLIED FROM SEGXREFI FILE
FABC0926E LOAD OF SDWCACA ATTEMPTED INTO A POPULATED DATABASE.
NUMBER OF SEGXREFO RECORDS GENERATED:      8
NUMBER OF LEVELS IN DATABASE HIERARCHY:      2

NAME      CODE    LVL    PAR    READ    CNT    ISRT    CNT    REPL    CNT    SELECTED OPTIONS
SDWCACA    1      1      0      0      1      0      0      0      0
SDWCAOA    2      2      1      0      0      0      0      0      0      REP-DUP
SDWCAAA    3      2      1      0      0      0      0      0      0      GHU-DUP
SDWCACO    4      2      1      0      0      0      0      0      0
SDWCACU    5      2      1      0      0      0      0      0      0
SDWCALT    6      2      1      0      0      0      0      0      0
SDWCALM    7      2      1      0      0      0      0      0      0      BYPASS
SDWCAEM    8      2      1      0      0      0      0      0      0
TOTAL:      0      0      0      0      1      0      0      0      0
END DFSRR00/FABCUR9
```

Figure 58. FABCUR9 Diagnostic Messages and Summary report (Case 2)

The following significant actions were taken:

- Message FABC0926E was returned, which specified that the database was already populated.
- Only the first record was processed
- Since the data set was not empty and REPL processing was not specified, the job did not end successfully, but condition code 8 was returned.

Case 3

The example shown in the following figure is the same as “Case 2” on page 103, except that the global action 'GHU' was specified, which shows that some REPL processing may be allowable.

```
DBDNAME=DATABAS1
OUTFILE
SEGXREFO
SEGXREFI
GHU
```

Figure 59. Control statements: Segment Cross-Reference table (Case 3)

The Diagnostic Messages and Summary report generated will be as in the following figure.

```

IMS HPFP UTILITIES - DEDBUR                "DEDB/HD UNLOAD DIAGNOSTIC MESSAGES AND SUMMARY REPORT"
5698-FPP                                DATE: 11/22/2020   TIME: 03.59.43                PAGE: 1
                                                    FABCUR9 - V2R1

FABC0900I CNTLCRDS:  DBDNAME=DATABAS1
FABC0900I CNTLCRDS:  OUTFILE
FABC0900I CNTLCRDS:  SEGXREFO
FABC0900I CNTLCRDS:  SEGXREFI
FABC0900I CNTLCRDS:  GHU

FABC0900I SEGXREFI: 001 001 000 SEGMENT1 V 00044 00000 00000
FABC0900I SEGXREFI: 002 002 001 SEGMENT2 V 00044 00000 00000 REPL
FABC0900I SEGXREFI: 003 002 001 SEGMENT3 V 00044 00000 00000 GHU
FABC0900I SEGXREFI: 004 002 001 SEGMENT4 V 00044 00000 00000
FABC0900I SEGXREFI: 005 002 001 SEGMENT5 V 00044 00000 00000
FABC0900I SEGXREFI: 006 002 001 SEGMENT6 V 00044 00000 00000
FABC0900I SEGXREFI: 007 002 001 SEGMENT7 V 00044 00000 00000 BYPASS
FABC0900I SEGXREFI: 008 002 001 SEGMENT8 V 00044 00000 00000

FORMAT OF UNLDREC: FP

NUMBER OF SEGMENT TYPES:      8
DBDNAME: DATABAS1
DBDNAME SUPPLIED FROM DURDBFN
SEGMENT NAMES SUPPLIED FROM SEGXREFI FILE
NUMBER OF SEGXREFO RECORDS GENERATED: 8
NUMBER OF LEVELS IN DATABASE HIERARCHY: 2

NAME      CODE  LVL  PAR  READ CNT  ISRT CNT  REPL CNT  SELECTED OPTIONS
SEGMENT1   1    1    0    1440      0        0      GHU-DUP
SEGMENT2   2    2    1      0      0        0      REP-DUP
SEGMENT3   3    2    1    1082      0        0      GHU-DUP
SEGMENT4   4    2    1      0      0        0      GHU-DUP
SEGMENT5   5    2    1      0      0        0      GHU-DUP
SEGMENT6   6    2    1      0      0        0      GHU-DUP
SEGMENT7   7    2    1    2849      0        0      BYPASS
SEGMENT8   8    2    1      83      0        0      GHU-DUP
TOTAL:     0    0    0    5454      0        0

```

Figure 60. FABCUR9 Diagnostic Messages and Summary report (Case 3)

The following significant actions were taken:

- The format in which the unload was expected was FP—the FPB DEDB Unload/Reload Utility format
- All of the records were read.
- At least GHU calls were issued for all segments receiving the 'II' status code from an ISRT. No DL/I calls were issued for SEGMENT7.
- REPL calls would have been issued for SEGMENT2; however, no SEGMENT2 data was included in the unload data set.
- No DL/I calls were issued for SEGMENT7.

Case 4

The example shown in the following figure is the same as “Case 3” on page 103, except that the global action 'REPL' is specified.

```

INPUT=FP
DBDNAME=DATABAS1
SEGXREFI
REPL

```

Figure 61. Control statements: Segment Cross-Reference table (Case 4)

The Diagnostic Messages and Summary report generated will be as in the following figure.


```
FABC0900I CNTLCRDS: INPUT=FP
FABC0900I CNTLCRDS: DBDNAME=DATBAS1
FABC0900I CNTLCRDS: SEGXREFI
FABC0900I CNTLCRDS: REPL

FABC0900I SEGXREFI: 001 001 000 SEGMENT1 V 00044 00000 00000
FABC0900I SEGXREFI: 002 002 001 SEGMENT2 V 00044 00000 00000 REPL
FABC0900I SEGXREFI: 003 002 001 SEGMENT3 V 00044 00000 00000 GHU
FABC0900I SEGXREFI: 004 002 001 SEGMENT4 V 00044 00000 00000
FABC0900I SEGXREFI: 005 002 001 SEGMENT5 V 00044 00000 00000
FABC0900I SEGXREFI: 006 002 001 SEGMENT6 V 00044 00000 00000
FABC0900I SEGXREFI: 007 002 001 SEGMENT7 V 00044 00000 00000 BYPASS
FABC0900I SEGXREFI: 008 002 001 SEGMENT8 V 00044 00000 00000
```

FORMAT OF UNLDREC: FP

NUMBER OF SEGMENT TYPES: 8
DBDNAME: DATBAS1
DBDNAME SUPPLIED FROM CNTLCRDS
SEGMENT NAMES SUPPLIED FROM SEGXREFI FILE
NUMBER OF LEVELS IN DATABASE HIERARCHY: 2

NAME	CODE	LVL	PAR	READ CNT	ISRT CNT	REPL CNT	SELECTED OPTIONS
SEGMENT1	1	1	0	1440	0	1440	REP-DUP
SEGMENT2	2	2	1	0	0	0	REP-DUP
SEGMENT3	3	2	1	1082	0	0	GHU-DUP
SEGMENT4	4	2	1	0	0	0	REP-DUP
SEGMENT5	5	2	1	0	0	0	REP-DUP
SEGMENT6	6	2	1	0	0	0	REP-DUP
SEGMENT7	7	2	1	2849	0	0	BYPASS
SEGMENT8	8	2	1	83	0	83	REP-DUP
TOTAL:	0	0	0	5454	0	1523	

Figure 62. FABCUR9 Diagnostic Messages and Summary report (Case 4)

The following significant actions were taken:

- The format in which the unload was expected was FP—the FPB DEDB Unload/Reload Utility format
- The database name DATBAS1, which is specified in the CNTLCRDS was used as the target database
- Replace had been specified for all segments receiving the 'II' status code from an ISRT except SEGMENT3, which was restricted to GHU, and SEGMENT 7, for which no DL/I calls were issued.

Case 5

In this example, control statements are specified as shown in the following figure.

```
DBDNAME=DATBAS1
OUTFILE
SEGXREFI
TEST
```

Figure 63. Control statements: Segment Cross-Reference table (Case 5)

The Summary report is shown in the following figure.

```
FABC0900I CNTLCRDS: DBDNAME=DATABAS1
FABC0900I CNTLCRDS: OUTFILE
FABC0900I CNTLCRDS: SEGXREFI
FABC0900I CNTLCRDS: TEST

FABC0900I SEGXREFI: 001 001 000 SEGMENT1 V 00044 00000 00000
FABC0900I SEGXREFI: 002 002 001 SEGMENT2 V 00044 00000 00000 REPL
FABC0900I SEGXREFI: 003 002 001 SEGMENT3 V 00044 00000 00000 GHU
FABC0900I SEGXREFI: 004 002 001 SEGMENT4 V 00044 00000 00000
FABC0900I SEGXREFI: 005 002 001 SEGMENT5 V 00044 00000 00000
FABC0900I SEGXREFI: 006 002 001 SEGMENT6 V 00044 00000 00000
FABC0900I SEGXREFI: 007 002 001 SEGMENT7 V 00044 00000 00000 BYPASS
FABC0900I SEGXREFI: 008 002 001 SEGMENT8 V 00044 00000 00000
```

FORMAT OF UNLDREC: FP

NUMBER OF SEGMENT TYPES: 8
DBDNAME: DATABAS1
DBDNAME SUPPLIED FROM CNTLCRDS
SEGMENT NAMES SUPPLIED FROM SEGXREFI FILE
NUMBER OF LEVELS IN DATABASE HIERARCHY: 2

NAME	CODE	LVL	PAR	READ CNT	ISRT CNT	REPL CNT	SELECTED OPTIONS
SEGMENT1	1	1	0	1440	0	0	TEST
SEGMENT2	2	2	1	0	0	0	REP-DUP TEST
SEGMENT3	3	2	1	1082	0	0	GHU-DUP TEST
SEGMENT4	4	2	1	0	0	0	TEST
SEGMENT5	5	2	1	0	0	0	TEST
SEGMENT6	6	2	1	0	0	0	TEST
SEGMENT7	7	2	1	2849	0	0	BYPASS TEST
SEGMENT8	8	2	1	83	0	0	TEST
TOTAL:	0	0	0	5454	0	0	

Figure 64. FABCUR9 Diagnostic Messages and Summary report (Case 5)

The following significant actions were taken:

- The format in which the unload was expected was FP (FPB DEDB Unload/Reload utility format)
- The database name DATABAS1, which is specified in the CNTLCRDS was used as the target database
- All other database format information was extracted from the Segment Cross-Reference Table
- No Database DL/I calls were issued
- Images of the segment data were written to the OUTFILE data set
- The Segment Cross-Reference table was rebuilt based upon data extracted from the SEGXREFI data set

Case 6

The example shown in the following figure is very similar to “Case 1” on page 102, except that an HD Reorganization Unload/Reload file was used as the source of the data.

The following CNTLCRDS were specified. An empty database was the target.

```
DBDNAME=DATABAS1
INPUT=FF
OUTFILE
SEGXREF0
SEGXREFI
```

Figure 65. Control statements: Segment Cross-Reference table (Case 6)

The Summary report is shown in the following figure.

```
FABC0900I CNTLCRDS: DBDNAME=DATABAS1
FABC0900I CNTLCRDS: INPUT=FF
FABC0900I CNTLCRDS: OUTFILE
FABC0900I CNTLCRDS: SEGXREFO
FABC0900I CNTLCRDS: SEGXREFI

FABC0900I SEGXREFI: 001 001 000 SEGMENT1 V 00044 00000 00000
FABC0900I SEGXREFI: 002 002 001 SEGMENT2 V 00044 00000 00000 REPL
FABC0900I SEGXREFI: 003 002 001 SEGMENT3 V 00044 00000 00000 GHU
FABC0900I SEGXREFI: 004 002 001 SEGMENT4 V 00044 00000 00000
FABC0900I SEGXREFI: 005 002 001 SEGMENT5 V 00044 00000 00000
FABC0900I SEGXREFI: 006 002 001 SEGMENT6 V 00044 00000 00000
FABC0900I SEGXREFI: 007 002 001 SEGMENT7 V 00044 00000 00000 BYPASS
FABC0900I SEGXREFI: 008 002 001 SEGMENT8 V 00044 00000 00000
NUMBER OF SEGMENT TYPES: 8
DBDNAME: DATABAS1 IMS LEVEL: 610
DBDNAME SUPPLIED FROM CNTLCRDS
SEGMENT NAMES SUPPLIED FROM SEGXREFI FILE
FORMAT OF UNLDREC: FF
NUMBER OF SEGXREFO RECORDS GENERATED: 8
NUMBER OF LEVELS IN DATABASE HIERARCHY: 2

NAME CODE LVL PAR READ CNT ISRT CNT REPL CNT SELECTED OPTIONS
SEGMENT1 1 1 0 1440 1440 0
SEGMENT2 2 2 1 0 0 0 REP-DUP
SEGMENT3 3 2 1 1082 1082 0 GHU-DUP
SEGMENT4 4 2 1 0 0 0
SEGMENT5 5 2 1 0 0 0
SEGMENT6 6 2 1 0 0 0
SEGMENT7 7 2 1 2849 0 0 BYPASS
SEGMENT8 8 2 1 83 83 0
TOTAL: 0 0 0 5454 2605 0
```

Figure 66. FABCUR9 Diagnostic Messages and Summary report (Case 6)

The following significant actions were taken:

- The format in which the unload was expected is FF—the HD Reorganization Unload/Reload file format.
- The database name DATABAS1, which is specified in the CNTLCRDS, was used as the target database.
- All other database format information was extracted from the Segment Cross Reference Table.
- REPL processing was allowed for SEGMENT2; however, since the database was empty, only Load or ISRT processing was applicable.
- Images of the segment data were written to the OUTFILE data set.
- The Segment Cross Reference table was rebuilt from data extracted from the SEGXREFI data set.

Example 4: Segment Cross-Reference files for segment format conversions

These are example of Segment Cross-Reference files for segment format conversions.

The following are examples of Segment Cross-Reference files for various segment format conversions.

Case 1

The following figure shows conversion from HD variable-length to HD fixed-length segment

```
001 001 000 SEGMENT1 V F 00040 00000 00000
002 002 001 SEGMENT2 V F 00040 00000 00000
003 002 001 SEGMENT3 V F 00040 00000 00000
004 002 001 SEGMENT4 V F 00040 00000 00000
```

Figure 67. Segment Cross-Reference file (Case 1)

Case 2

The following figure shows conversion from HD fixed-length to HD variable-length segment

```
001 001 000 SEGMENT1 F V 00040 00000 00000
002 002 001 SEGMENT2 F V 00040 00000 00000
003 002 001 SEGMENT3 F V 00040 00000 00000
004 002 001 SEGMENT4 F V 00040 00000 00000
```

Figure 68. Segment Cross-Reference file (Case 2)

Case 3

The following figure shows conversion from DEDB fixed length to HD fixed length segment

```
001 001 000 SEGMENT1 F F 00040 00000 00000
002 002 001 SEGMENT2 F F 00040 00000 00000
003 002 001 SEGMENT3 F F 00040 00000 00000
004 002 001 SEGMENT4 F F 00040 00000 00000
```

Figure 69. Segment Cross-Reference file (Case 3)

Case 4

The following figure shows conversion from HD fixed length to DEDB fixed length segment

```
001 001 000 SEGMENT1 F F 00040 00000 00000
002 002 001 SEGMENT2 F F 00040 00000 00000
003 002 001 SEGMENT3 F F 00040 00000 00000
004 002 001 SEGMENT4 F F 00040 00000 00000
```

Figure 70. Segment Cross-Reference file (Case 4)

Case 5

The following figure shows conversion from DEDB fixed-length to HD variable-length segment

```
001 001 000 SEGMENT1 F V 00040 00000 00000
002 002 001 SEGMENT2 F V 00040 00000 00000
003 002 001 SEGMENT3 F V 00040 00000 00000
004 002 001 SEGMENT4 F V 00040 00000 00000
```

Figure 71. Segment Cross-Reference file (Case 5)

Case 6

The following figure shows conversion from HD variable-length to DEDB fixed-length segment

```
001 001 000 SEGMENT1 V F 00040 00000 00000
002 002 001 SEGMENT2 V F 00040 00000 00000
003 002 001 SEGMENT3 V F 00040 00000 00000
004 002 001 SEGMENT4 V F 00040 00000 00000
```

Figure 72. Segment Cross-Reference file (Case 6)

Case 7

The following figure shows conversion from DEDB variable-length to HD fixed-length segment

```
001 001 000 SEGMENT1 V F 00040 00000 00000
002 002 001 SEGMENT2 V F 00040 00000 00000
003 002 001 SEGMENT3 V F 00040 00000 00000
004 002 001 SEGMENT4 V F 00040 00000 00000
```

Figure 73. Segment Cross-Reference file (Case 7)

Case 8

The following figure shows conversion from HD fixed-length to DEDB variable-length segment

```
001 001 000 SEGMENT1 F V 00040 00000 00000
002 002 001 SEGMENT2 F V 00040 00000 00000
003 002 001 SEGMENT3 F V 00040 00000 00000
004 002 001 SEGMENT4 F V 00040 00000 00000
```

Figure 74. Segment Cross-Reference file (Case 8)

Case 9

The following figure shows conversion from DEDB fixed-length to DEDB variable-length segment

001	001	000	SEGMENT1	F	V	00040	00000	00000
002	002	001	SEGMENT2	F	V	00040	00000	00000
003	002	001	SEGMENT3	F	V	00040	00000	00000
004	002	001	SEGMENT4	F	V	00040	00000	00000

Figure 75. Segment Cross-Reference file (Case 9)

Case 10

The following figure shows conversion from DEDB variable-length to DEDB fixed-length segment

001	001	000	SEGMENT1	V	F	00040	00000	00000
002	002	001	SEGMENT2	V	F	00040	00000	00000
003	002	001	SEGMENT3	V	F	00040	00000	00000
004	002	001	SEGMENT4	V	F	00040	00000	00000

Figure 76. Segment Cross-Reference file (Case 10)

Related reference

CNTLCRDS DD data set

The CNTLCRDS data set contains the user's description of the database conversion processing to be done by FABCUR9.

Chapter 8. IMS DEDB randomizing module

The IMS DEDB randomizing module interface module is named FABCRMIF. FABCRMIF supports the XCI randomizer interface for batch.

FABCRMIF has the alias name FABARMIF and FABBRMIF. You can invoke FABCRMIF by using these alias names.

FABCRMIF provides an easy-to-use facility for invoking a DEDB randomizer from an application program.

Using FABCRMIF, an application program can request that RAP values be calculated for up to 16 different DBD names.

Module FABDRMIF provides the same facility as FABCRMIF but has a different interface for the CALC call than the call has with FABCRMIF. FABDRMIF does not have the fourth parameter (UOW number) as FABCRMIF does.

This facility is of greatest use when the calling program is coded in a high-level language, but it is usable from programs written in any language conforming to the register linkage conventions of IBM z/OS.

Topics:

- [“DEDB Unload/Reload RAP number to RAP RBA conversion” on page 111](#)
- [“Interface parameter” on page 112](#)
- [“Link-editing your program” on page 114](#)
- [“Required JCL DD statements to run your program” on page 114](#)
- [“Input for the IMS DEDB randomizing module” on page 115](#)
- [“IMS DBT 2.x application that process multiple DEDBs sequentially” on page 116](#)

DEDB Unload/Reload RAP number to RAP RBA conversion

This topic explains the RAP number to RAP RBA conversion for the FPA Unload, Reload, and Change functions, and the FPB DEDB Unload and Reload utilities.

This topic contains product sensitive programming interface information.

PSPI

This topic gives the details for the RAP number to RAP RBA conversion. This RAP number to RAP RBA conversion is a product-sensitive programming interface for the customer. The IMS interface defined for DEDB randomizing routines specifies, upon return from the randomizer, that:

- Register 1 will contain the address of the DMAC for the area selected.
- Register 0 will contain the "relative RAP number" within that area to which the root segment is assigned.

The formula to convert that "relative RAP number" to the RBA of that RAP is as follows:

```
RAP RBA = CIsiz * [ INT( RAP# / (UOW1 - UOW2) ) * UOW1
                  + REM( RAP# / (UOW1 - UOW2) ) + 2 ]
```

where:

RAP# = Relative RAP number (from randomizer)

UOW1 = UOW part 1

UOW2 = UOW part 2

INT(x/y) = Integer value of the quotient from the division of 'x' by 'y'

REM(x/y) = Remainder from the division of 'x' by 'y'

Note: This randomizer is invoked with the key value of the root segment for a database record. The RAP RBA value from that call should be saved for use by the records of the FPA Unload, Reload, and Change functions, and the FPB DEDB Unload and Reload utilities for all dependent segments of that root.

PSPI

Interface parameter

This topic describes the interface parameters of the IMS DEDB randomizing module.

This topic contains product sensitive programming interface information.

PSPI

FABCRMIF contains three functional units: initialization, to invoke the randomizing module, and to end the XCI randomizer.

An application program must call the FABCRMIF routine with the initialization parameter first to establish the environment for invoking the randomizing module. Then the application calls FABCRMIF, using the randomizing module invocation parameter for each root segment key.

To invoke the XCI randomizer, an application program must call the FABCRMIF routine with the termination parameter to terminate the processing of the associated DBD. If the application does not call the FABCRMIF with the termination parameter in the XCI randomizer environment, the resources obtained by the XCI randomizer at the initialization call will not be released until the end of the application.

Initialization parameter

FABCRMIF is called for initialization using the following parameter list:

Function code

This is an optional parameter. It must be a 4-byte character field containing INIT.

The initialization function sets the required environment for subsequent calculation (CALC) calls. The INIT function must be performed prior to a CALC call of the associated DBD. Up to 16 INIT calls can be made for 16 different DBD names.

In the case of the XCI randomizer routine, up to 16 DBDs can be processed by calling the TERM call.

DBD name

This is a mandatory parameter. It must be an 8-byte character field containing the DBD name of the DEDB being processed (that is, the member name of ACBLIB).

GDD feedback area

This is an optional parameter. It is a 32-byte field to receive a copy of the control block built by the "Get DEDB DMB" subroutine that is invoked during randomizing module interface initialization. You can use macro FABAMGDD, FABBMGDD, or FABCMGDD to generate or map the internal control block in assembly language.

The GDDP@DDT field of the GDD feedback area contains the address of the Database Description Table (DDT) built by the "Get DEDB DMB" subroutine. You can use macro FABAMDDT, FABBMDDT, or FABCMDDT to map the DDT in assembly language.

Randomizing module name

This is an optional parameter. It is an 8-byte field containing a member name of the load module in the RMODLIB data set to be used as the randomizer for this execution (instead of the module named in the DMB read from ACBLIB). If this parameter is not specified, the randomizer named in the DMB read from ACBLIB is used.

Randomizing module invocation parameter

FABCRMIF is called for invocation of the randomizing module using the following parameter list:

Function code

This is a mandatory parameter. It must be a 4-byte character field containing CALC.

Root segment key

This is a mandatory parameter. Field length is as defined in the DMB specified as the DBD name for the INIT parameter. This parameter contains the value of the key for a root segment. This is the data used by the randomizing module to calculate area number and RAP number values.

Area number

This is a mandatory parameter. It must be a 2-byte binary field. In this field, the randomizing module will return the area number calculated using the root segment key.

RAP RBA

This is a mandatory parameter. It must be a 4-byte binary field. In this field, the randomizing module will return the RAP RBA calculated using the root segment key.

UOW number

This is an optional parameter. It must be a 4-byte binary field. In this field, the randomizing module will return the UOW number calculated using the root segment key.

DBD name

This is an optional parameter. It must be an 8-byte character field containing the DBD name of the DEDB being processed (that is, the member name of ACBLIB).

If this parameter is not specified, then the DBD name specified in the first INIT call is performed.

Note: The high-order bit of the last address parameter must be set to 1 the bit can be checked to find the end of the list.

The following figure is an example of code written in assembly language.

```

      .
      .
      .
      CALL  FABCRMIF, (FCDINIT,DBDNAME,GDDFBKA,RMODNAME),VL
      .
      .
      CALL  FABCRMIF, (FDCALC,ROOTKEY,AREANO,RAPRBA,UOWNO),VL
      .
      .
      FCDINIT DC    CL4'INIT'          /* FUNCTION CODE 'INIT'
      DBDNAME DC    CL8'VRSDSRF'       /* DBD NAME
      RMODNAME DC   CL8'RMOD4'         /* RANDOMIZING MODULE NAME
      .
      .
      FDCALC  DC    CL4'CALC'          /* FUNCTION CODE 'CALC'
      ROOTKEY DS    XL10'00'          /* ROOT KEY AREA
      AREANO  DS    H                  /* AREA NUMBER (RETURN)
      UOWNO   DS    F                  /* UOW NUMBER (RETURN)
      RAPRBA  DS    F                  /* RAP RBA (RETURN)
      .
      .
      GDDFBKA FABCMGDD DSECT=NO,LIST=YES /* GDD FEED BACK AREA
      .
      .
      .
```

Figure 77. Sample program for invoking an IMS DEDB randomizing module written in assembly language

Termination parameter

FABCRMIF is called for the XCI randomizer termination by using the following parameter list:

Function code

This is a mandatory parameter for the XCI randomizer. It must be a 4-byte character field containing "TERM". The TERM function must be run for the DBD that has issued an INIT call.

DBDNAME

This is a mandatory parameter. It must be an 8-byte character field containing the DBD name of the DEDB that has been processed (that is, the member name of ACBLIB).

PSPI

Link-editing your program

This topic describes how to link-edit your program with the IMS DEDB randomizing module.

This topic contains product sensitive programming interface information.

PSPI

FABCRMIF is link-edited into a user program or can be invoked dynamically by use of ATTACH, LINK, or DYNAMIC calls. The IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0) must be concatenated to your application program library in the JOBLIB/STEPLIB DD statement.

There is no restriction regarding application's run mode (AMODE and RMODE) to link-edit or to invoke the FABCRMIF program.

PSPI

Required JCL DD statements to run your program

This topic describes the required JCL DD statements to run your program with the IMS DEDB randomizing module.

This topic contains product sensitive programming interface information.

PSPI

There are several DD statements that you must specify to call the FABARMIF routine.

The following table shows these DD statements.

Table 15. FABARMIF DD statements

DDNAME	Use	Format	Required or optional
JOBLIB/STEPLIB	Input	PDS	Required
ACBLIB	Input	PDS	Optional
IMSACBA	Input	PDS	Optional
IMSACBB	Input	PDS	Optional
MODSTAT	Input		Optional
MODSTAT2	Input		Optional
RMODLIB	Input	PDS	Optional
RMIFCTL	Input	LRECL=80	Optional

JOBLIB/STEPLIB DD

Defines the library that contains your application program.

The following libraries must be concatenated to your application program library in the JOBLIB/STEPLIB DD statement:

- The IMS HP Fast Path Utilities load module library (HPFP.SHFPLMD0).

- The IMS Tools Base library (SGLXLOAD), if you specify the IMSCATHLQ=*bsdshlq* keyword.

ACBLIB DD

Defines the library that contains the DMB for the database. This DD statement must be provided when IMSCATHLQ=*NO.

If MODSTAT/MODSTAT2 DD is provided, this DD is not necessary.

If you specify the IMSCATHLQ=*bsdshlq* keyword, ACBLIB DD statement is not used. The IMS directory is used instead of the ACB library.

IMSACBA DD

Defines the library that contains the DMB for the database. This DD statement must be provided if MODSTAT/MODSTAT2 DD is specified.

IMSACBB DD

Defines the library that contains the DMB for the database. This DD statement must be provided if MODSTAT/MODSTAT2 DD is specified.

MODSTAT DD

Defines the MODSTAT data set. When this DD statement is specified, the IMSACBA and IMSACBB DD statements must be specified instead of the ACBLIB DD.

MODSTAT2 DD

Defines the MODSTAT data set. When this DD statement is specified, the IMSACBA and IMSACBB DD statements must be specified instead of the ACBLIB DD.

RMODLIB DD

Defines the library in which the randomizing routine(s) resides.

Instead of defining the library on this DD statement, the library can be concatenated on the JOBLIB/STEPLIB DD statement.

RMIFCTL DD

Defines the control statement input data set. This data set can reside on a direct-access device, or be routed through the input stream. See [“RMIFCTL DD data set” on page 115](#).

PSPI

Input for the IMS DEDB randomizing module

Use the following information to supply RMIFCTL control statements.

RMIFCTL DD data set

The RMIFCTL data set contains control statements to enable IMS-managed ACBs for the FABCRMIF and FABDRMIF modules.

Control statement syntax

- Control statements are coded on 80-byte records.
- All control statement specifications must start in column 1. A control statement record can include only one control statement.
- A "keyword=value" specification may not span the control statement.

RMIFCTL control statements

The following statements are supported for the RMIFCTL DD data set:

- IMSCATHLQ
- IMSCATACB_INPUT

IMSCATHLQ control statement

The optional IMSCATHLQ statement specifies the high-level qualifier of the bootstrap data set of the IMS directory, which is an extension of the IMS catalog. You must enable the IMS catalog and the IMS management of ACBs when you specify the high-level qualifier of the bootstrap data set of the IMS directory.

```
IMSCATHLQ={*NO | bsdshlq}
```

bsdshlq

Reads the ACB member from the IMS directory instead of the ACB library by using IMS Tools Catalog Interface. *bsdshlq* specifies the high-level qualifier of the IMS directory bootstrap data set.

***NO**

Reads the ACB member from the ACB library. IMSCATHLQ=*NO is the default value.

IMSCATACB_INPUT control statement

The optional IMSCATACB_INPUT statement specifies whether to retrieve the currently active ACB definition or the pending ACB definition from the IMS directory. This statement is effective only when the IMSCATHLQ=*bsdshlq* statement is specified.

```
IMSCATACB_INPUT={CURRENT | PENDING}
```

CURRENT

The currently active ACB member is retrieved from the IMS directory data sets. IMSCATACB_INPUT=CURRENT is the default value.

PENDING

The pending ACB member is retrieved from the staging data set.

IMS DBT 2.x application that process multiple DEDBs sequentially

For an IMS DBT 2.x application that issues multiple combinations of INIT and CALC calls to process multiple DEDBs sequentially, FABDRMIO must be link-edited into the application instead of FABARMIF, FABBRMIF, or FABCRMIF.

In a CALC call for FABDRMIO, the sixth parameter "DBD name" is not effective even if you specify it, and the CALC call is made for the DEDB that was specified in the INIT call issued just before the CALC call.

The TERM call for the XCI randomizer is effective the same way for FABARMIF, FABBRMIF, FABCRMIF, and FABDRMIF.

Chapter 9. Standard format extract data interface module

These reference topics describe the standard format extract data interface, which is an I/O service module for reading data from sequential files that are created by the Extract function of FPA and the ODE utility of FPO.

Topics:

- [“Overview of the standard format extract data interface” on page 117](#)
- [“Using the standard format extract data interface” on page 118](#)
- [“Output report of standard format extract data interface” on page 126](#)

Overview of the standard format extract data interface

The standard format extract data interface (FPXGXDR0 or its alias FABGXDR) is an I/O service module that can be called by user-written programs to read data from sequential files created by the Extract process of FPA or by the ODE utility of FPO.

By using the standard format extract data interface service module as an interface, the user-written application program is insulated from future changes to the standard format of the extract function. If, in the future, the Extract function should change the format of its output records or files, standard format extract data interface will adapt to the changed format while presenting an unchanged interface to the user. The application program using standard format extract data interface should be able to run unchanged, thus minimizing programming changes.

Standard format extract data interface generates a statistics report file detailing the number and type of segments read from each file. This report can be compared with reports generated by the Extract function for verification of the statistics.

Modes of operation

Standard format extract data interface has two modes of operation; single file mode and multiple file mode. The single file mode is somewhat simpler to use, but the multiple file mode offers you the flexibility of having more than one input file. You might want to do this if you want to process data from multiple runs of the Extract function.

Single file mode

Use this mode if you have only one input file. The input file can be in STD format. Using the single file mode, the data in the input file is accessed with INIT, GET and EOJ calls. The EOJ call must be issued before the program ends.

Multiple file mode

Use this mode for processing up to nine input files concurrently. Each input file can be from a different database and can be in either SORT or STD format.

The caller identifies the file for each operation by appending a suffix of 1 - 9 to each service request: INIx, GETx, and EOJx. The suffix x corresponds to the suffix on the DD statement for the file. The EOJx call must be issued before the program ends.

Related reference

[Specifying JCL for the standard format extract data interface](#)

In order to run the standard format extract data interface, you need to look at your STEPLIB statement, add some DD DATA statements, and optionally add a SYSOUT statement.

Using the standard format extract data interface

In order to use the standard format extract data interface, you need to define a parameter list that can be passed to the interface module, and review and possibly modify your JCL.

Parameter list of the standard format extract data interface

The standard format extract data interface is called with a parameter list consisting of one to three parameters.

All three parameters are required for GET or GETx calls.

Only the first parameter is required for INIT, INIx, EOJ, and EOJx calls.

Parameter 1

The purpose of the first parameter is to specify the function code (call) to the interface. In addition, the parameter provides fields in which standard format extract data interface returns the segment code and segment name and a status code related to the record is processes.

The following figure shows the layout of the standard format extract data interface for parameter 1.

```
01 PARMLIST.  
  05 FUNCTION.  
    10 FUNC3    PIC X(3).  
    10 SUFFIX    PIC X.  
  05 STATUS      PIC X(2).  
  05 SEGCODE     PIC S9(4) COMP.  
  05 SEGNAME     PIC X(8).
```

Figure 78. Standard format extract data interface parameter 1 layout

Valid function codes are:

- GET or GETx
- INIT or INIx
- EOJ or EOJx

Status codes that might be returned are:

''

Call successful

'GB'

End of data

'EA'

End of data for one area (concatenated input or run-level file).

The segment code and segment name are returned from FPXGXDR0 and are not input parameters.

Parameter 2

This parameter defines the key length and the key area. As mentioned, this parameter is only used with GET and GETx calls.

The following figure shows the layout of the standard format extract data interface for parameter 2.

```
01 KEYPARM  
  05 KEYLEN      PIC S9(4) COMP.  
  05 KEYAREA     PIC X(3840).
```

Figure 79. Standard format extract data interface parameter 2 layout

Note that if an SDEP segment is extracted, KEYLEN will contain a value of zero. KEYAREA will be kept as is.

Parameter 3

This third parameter has two fields, one for the length of the data, the other for the data itself.

This parameter is only used with GET and GETx calls.

The following figure shows the layout of the standard format extract data interface for parameter 3.

```
01 DATAPARM
05 DATALEN      PIC S9(4) COMP.
05 DATAAREA     PIC X(28552).
```

Figure 80. Standard format extract data interface parameter 3 layout

Note that if a full segment is extracted, DATAAREA will contain the LL field of the segment followed by the segment data. DATALEN is equal to the LL field.

Specifying JCL for the standard format extract data interface

In order to run the standard format extract data interface, you need to look at your STEPLIB statement, add some DD DATA statements, and optionally add a SYSOUT statement.

STEPLIB statement

The STEPLIB on the calling job step must refer to the library containing the standard format extract data interface load module, FPXGXDR0.

DD DATA statements

Add DD DATA statements that define the input data set(s) containing the data extracted by the Extract function. The input file might be a file containing extracted data from more than one area. These input files can be in STD format.

Like-formatted files from the same database can be concatenated.

PRINT data set

You can specify XDRPRINT as the DD name, and be sure to specify RECFM=FBA and LRECL=121. Alternatively, you can specify a JES spool file.

If you do not include an XDRPRINT statement, output is discarded. So it is strongly recommended you include an XDRPRINT statement.

Example JCL

The following figure provides an example of JCL statements supporting standard format extract data interface. See this example when modifying your JCL.

```
//STEPLIB DD DISP=SHR,DSN=User.library      <-- User-written program
//          DD DISP=SHR,DSN=HPFP.SHFPLMD0    <-- PGMLIB concatenated
//XDRPRINT DD SYSOUT=X
//XDRDATA1 DD DISP=SHR,DSN=FPX.ODE.DATABASE.STD <-- Input data set
```

Figure 81. Standard format extract data interface example JCL

Invoking the standard format extract data interface

The standard format extract data interface can be called from your program with PL/I, assembly, and COBOL languages.

The following examples illustrate calling standard format extract data interface. These are just examples; your invocation of the utility will differ somewhat to reflect your situation.

Subsections:

- [“Invocation with PL/I” on page 120](#)
- [“Invocation with assembly language” on page 121](#)
- [“Invocation with COBOL” on page 124](#)

Invocation with PL/I

The following figure shows PL/I Invocation of the standard format extract data interface.

```

SAMPGXRP: PROCEDURE  OPTIONS (MAIN) REORDER
/*****
/*
/* SAMPGXDP
/*
/* SAMPLE DRIVER FOR FPXGXDR0 USING PL/I
/*
/*
*****/
0 DEFAULT RANGE(*) STATIC;

DCL ADDR          BUILTIN;
DCL RETCODE       FIXED BIN(31);
DCL PLIRETC       BUILTIN;

DCL 1 CALLS,
    3 GET_VAR CHAR(4) INIT ('GET1'),
    3 INI_VAR  CHAR(4) INIT ('INI1'),
    3 EOJ_VAR  CHAR(4) INIT ('EOJ1');

DCL 1 PARM1,
    3 P1FUNC CHAR(4), /* 3 CHAR FUNC, 1 CHAR SUFFIX */
    3 P1STAT CHAR(2),
    3 P1SEGC FIXED BIN(15),
    3 P1SEGN CHAR(8);

DCL PARM2          POINTER;
DCL PARM2_AREA CHAR(32760) CONTROLLED;
DCL PARM3          POINTER;
DCL PARM3_AREA CHAR(32760) CONTROLLED;

DCL FPXGXDR0      ENTRY OPTIONS(ASSEMBLY INTER);
DCL ABEND         ENTRY OPTIONS(ASSEMBLY INTER); /* ANY ABEND ROUTINE */

RETCODE = 0;

ALLOCATE PARM2_AREA;
ALLOCATE PARM3_AREA;

PARM2 = ADDR(PARM2_AREA);
PARM3 = ADDR(PARM3_AREA);

P1FUNC = INI_VAR;
P1STAT = 'XX';

```

Figure 82. PL/I invocation of standard format extract data interface (Part 1 of 2)


```

CALL FPXGXDR0(PARM1)
                                /* INIT CALL    */

IF P1STAT = ' '
THEN
DO;
    P1FUNC = GET_VAR;
    P1STAT = 'XX';

    CALL FPXGXDR0(PARM1,PARM2_AREA,PARM3_AREA);

    DO WHILE (P1STAT = ' ' |
              P1STAT = 'EA');
                                /* MORE DATA    */
                                /* END OF 1 AREA */

        P1FUNC = GET_VAR;
        P1STAT = 'XX';

        CALL FPXGXDR0(PARM1,PARM2_AREA,PARM3_AREA);
    END;

    IF P1STAT = 'GB'
    THEN
    DO;
        P1FUNC = EOJ_VAR;
        P1STAT = 'XX';
        CALL FPXGXDR0(PARM1);
                                /* EOJ CALL      */
    END;

END;

IF P1STAT ^= ' '
THEN CALL ABEND;

CALL PLIRETC(RETCODE);

END SAMPGXRP;

```

Figure 83. PL/I invocation of standard format extract data interface (Part 2 of 2)

Invocation with assembly language

The following figure is an example of an assembly code invocation of the standard format extract data interface.

```

*****
*                                                                 *
* SAMPGXDA                                                         *
*                                                                 *
*   Sample driver for FPXGXDR0 using Assembler language          *
*                                                                 *
*                                                                 *
*****
SAMPGXDA CSECT
SAMPGXDA AMODE 24
SAMPGXDA RMODE 24
SAMPGXDA FPXENTRY
*
*   L      15,=V(FPXGXDR0)
*   ST      R15,EPAGXDR
*
*   OPEN    (DCB0,OUTPUT)
*   OPEN    (DCB1,OUTPUT)
*   OPEN    (DCB2,OUTPUT)
*
*   L      R0,=F'3844'
*   STORAGE OBTAIN,LENGTH=(R0)
*   ST      R1,KEYWORK
*
*   L      R0,=F'28556'
*   STORAGE OBTAIN,LENGTH=(R0)
*   ST      R1,DATAWORK
*
*   L      R0,=F'28556'
*   STORAGE OBTAIN,LENGTH=(R0)
*   ST      R1,WORKAREA
*
*   MVC     P1FUNC(4),=C'INI1'
*   MVC     P1STAT(2),=C'XX'
*
*   L      R15,EPAGXDR
*   CALL    (15),(PARM1),VL
*
*
*   IF      (CLC,P1STAT,NE,=C' ')
*       ABEND 100,DUMP
*   ENDIF
*
*   DO INF
*
*       MVC     P1STAT,=CL2'XX'
*
*       MVC     P1FUNC(4),=C'GET1'
*       MVC     P1STAT(2),=C'XX'
*
*
*   LA      R1,PARMLIST
*   OI      DATAWORK,X'80' *

```

Figure 84. Assembly code invocation of standard format extract data interface (Part 1 of 3)

```

L      R15,EPAGXDR
BALR   R14,R15
*
DOEXIT (CLC,P1STAT,EQ,=C'GB')
*
IF  (CLC,P1STAT,NE,=C'  '),ANDIF,(CLC,P1STAT,NE,=C'GB'),          X
    ANDIF,(CLC,P1STAT,NE,=C'EA')
    ABEND 150,DUMP
ENDIF
*
IF  (CLC,P1STAT,EQ,=C'EA')
*
    WTO 'END OF AREA INDICATED',ROUTCDE=11
*
ELSE
*
    L      R3,WORKAREA
    L      R10,KEYWORK
    LH     R1,0(R10)
    LA     R1,4(R1)
    STH    R1,0(R3)
    XC     2(2,R3),2(R3)
    LA     R14,2(R10)
    LR     R15,R1
    LA     R0,4(R3)
    MVCL   R0,R14
*
    PUT    DCB0,0(R3)
*
    L      R3,WORKAREA
    L      R10,DATAWORK
    LH     R1,0(R10)
    LA     R1,4(R1)
    STH    R1,0(R3)
    XC     2(2,R3),2(R3)
    LA     R14,2(R10)
    LR     R15,R1
    LA     R0,4(R3)
    MVCL   R0,R14
*
    PUT    DCB1,0(R3)
*
    MVI    IMAGE,C'  '          CLEAR 1st byte
    MVC    IMAGE+1(79),IMAGE    PROPAGATE
    MVC    IMAGE(8),P1SEGN
    MVC    IMAGE+9(2),P1SEGC
*
    PUT    DCB2,IMAGE
*
ENDIF
*
ENDDO

```

Figure 85. Assembly code invocation of standard format extract data interface (Part 2 of 3)

```

*
MVC      P1FUNC(4),=C'E0J1'
L        R15,EPAGXDR
CALL     (15),(P1FUNC),VL
*
IF (CLC,P1STAT,NE,=C' ')
  ABEND 199,DUMP
ENDIF
*
CLOSE DCB0
CLOSE DCB1
CLOSE DCB2
*
FPXLEAVE
*
IMAGE    DS      CL80' '
*
EPAGXDR  DS      F
*
PARMLIST DC      AL4(PARM1)
KEYWORK  DS      F
DATAWORK DS      F
         DC      CL8'WORKAREA'
WORKAREA DS      F
*
DCB0      DCB    DSORG=PS,MACRF=PM,DDNAME=SAMPKEY,RECFM=VB,LRECL=3844
DCB1      DCB    DSORG=PS,MACRF=PM,DDNAME=SAMPDATA,RECFM=VB,LRECL=28556
DCB2      DCB    DSORG=PS,MACRF=PM,DDNAME=SYSPRINT,RECFM=FB,LRECL=80
*
*
PARM1     EQU      *
P1FUNC    DS      CL3
P1SUFIX   DS      CL1
P1STAT    DS      CL2
P1SEGC    DS      XL2
P1SEGN    DS      CL8
*
LTORG
END

```

Figure 86. Assembly code invocation of standard format extract data interface (Part 3 of 3)

Invocation with COBOL

The following figure is an example of a COBOL program calling the standard format extract data interface.

IDENTIFICATION DIVISION.	00010000
PROGRAM-ID. SAMCOB.	00020000
ENVIRONMENT DIVISION.	00030000
DATA DIVISION.	00040000
WORKING-STORAGE SECTION.	00050000
*	00080000
*****	00080011
*	00080012
* SAMPGXRC	00080013
*	00080014
* Sample driver for FPXGXDR0 using COBOL	00080015
*	00080016
*	00080017
*****	00080018
*	00080200
01 CALLS.	00090000
03 GET-VAR PIC X(4) VALUE 'GET1'.	00100000
03 INI-VAR PIC X(4) VALUE 'INI1'.	00110000
03 EOJ-VAR PIC X(4) VALUE 'EOJ1'.	00120000
*	00130000
*	00132003
01 PARM1.	00140000
03 P1FUNC PIC X(4).	00150000
03 P1STAT PIC X(2).	00160000
03 P1SEGC PIC S9(04) COMP.	00170016
03 P1SEGN PIC X(8).	00180000
*	00190000
01 PARM2-AREA PIC X(32760).	00210021
*	00230000
01 PARM3-AREA PIC X(32760).	00240022
*	00250021
PROCEDURE DIVISION.	00260000
ENTRY 'FPXGXDR0'.	00270120
ENTRY 'ABEND'.	00271006
*	00280000
MOVE INI-VAR TO P1FUNC.	00300005
MOVE 'XX' TO P1STAT.	00310005
*	00320000
CALL 'FPXGXDR0' USING PARM1.	00330016
IF P1STAT = ' '	00340010
MOVE GET-VAR TO P1FUNC	00350005
MOVE 'XX' TO P1STAT	00360005
*	00362017
CALL 'FPXGXDR0' USING PARM1, PARM2-AREA, PARM3-AREA	00380020
*	00390001
PERFORM UNTIL P1STAT NOT EQUAL ' ' AND P1STAT EQUAL 'EA'	00400023
*	00410117
MOVE GET-VAR TO P1FUNC	00411017
MOVE 'XX' TO P1STAT	00412017
*	00413117
CALL 'FPXGXDR0' USING PARM1, PARM2-AREA, PARM3-AREA	00414020
	00420011
END-PERFORM	00430018
END-IF.	00440018
	00450017
IF P1STAT = 'GB'	00470011
MOVE GET-VAR TO P1FUNC	00480005
MOVE 'XX' TO P1STAT	00490011
CALL 'FPXGXDR0' USING PARM1	00491016
END-IF.	00520012
*	00530000
IF P1STAT NOT = SPACE	00540009
CALL 'ABEND'	00550009
END-IF.	00560009
*	00570000
EOJ-EXIT.	00610117
MOVE ZERO TO RETURN-CODE.	00611017
GOBACK.	00620017

Figure 87. COBOL invocation of standard format extract data interface

Output report of standard format extract data interface

Output from standard format extract data interface consists of a report written to data set XDRPRINT. This report shows the number and types of segments read from each input file.

The XDRPRINT output file is also used for error messages. If errors occur when calling the standard format extract data interface utility, perhaps due to invalid parameters or invalid file suffixes, a message is written to data set XDRPRINT and ABEND U3599 occurs.

The XDRPRINT DD statement is mandatory.

For error messages, see the topic "FPX messages" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

The following figure shows a report showing the number and type of segments read from each file.

```
IMS HPFP UTILITIES                                "ONLINE DATA EXTRACT INTERFACE OUTPUT"
5698-FPP                                           DATE: 11/22/2020 13:28:52
Statistics for DDNAME XDRDATA1.
Number of areas processed = 00012.
Number of segment code 001 segment name ROOTSEG1 segments read = 00000040
Number of segment code 002 segment name SDSEGNM1 segments read = 00000000
Number of segment code 003 segment name DD1      segments read = 00000006
Number of segment code 004 segment name DD2      segments read = 00000004
Number of segment code 005 segment name DD3      segments read = 00000005
Number of segment code 006 segment name DD4      segments read = 00000007
Number of segment code 007 segment name DD5      segments read = 00000011
Number of segment code 008 segment name DD6      segments read = 00000005
Total records read = 00000078
PAGE: 1
ODE - V2.R1
```

Figure 88. Standard Format Extract Data Interface Output report

Chapter 10. Integration with IMS HP Image Copy

IMS HP Fast Path Utilities provides the HPFPU HASH Check support and the DB Sensor support for IMS HP Image Copy. These capabilities can be invoked within the IMS HP Image Copy job step.

Topics:

- [“HPFPU Hash Check support for IMS HP Image Copy” on page 127](#)
- [“DB Sensor support for IMS HP Image Copy” on page 132](#)

HPFPU Hash Check support for IMS HP Image Copy

IMS HP Image Copy can call the HPFPU Hash Check support. The HPFPU Hash Check support is invoked during the image copy processing of an IMS HP Image Copy job.

Topics:

- [“Input for HPFPU Hash Check support for IMS HP Image Copy” on page 127](#)
- [“Output for HPFPU Hash Check support for IMS HP Image Copy” on page 127](#)
- [“Examples: HPFPU Hash Check support for IMS HP Image Copy” on page 131](#)

Input for HPFPU Hash Check support for IMS HP Image Copy

To run an IMS HP Image Copy job for DEDB area data sets, you must specify additional DD statements and some keywords in the ICEIN control statements.

The HPFPU Hash Check support is invoked for each area that is specified by the DEDBPC=Y keyword in the utility control statement. The JCL requirements for using the HPFPU Hash Check support within an IMS HP Image Copy job are described in the *IMS High Performance Image Copy User's Guide*.

Output for HPFPU Hash Check support for IMS HP Image Copy

The following topics describe the output produced by the HPFPU Hash Check support for IMS HP Image Copy.

For more information, see the *IMS High Performance Image Copy User's Guide*.

MSGOUT DD data set

This data set contains the messages that are issued by the HPFPU Hash Check support for IMS HP Image Copy.

For more information, see the *IMS High Performance Image Copy User's Guide*.

REPORTS DD data set

This data set contains the DEDB Area Analysis reports, the Segment Length Distribution report, and the Process Summary report.

For more information, see the *IMS High Performance Image Copy User's Guide*.

Subsections:

- [“DEDB Area Analysis reports” on page 128](#)
- [“Segment Length Distribution report” on page 128](#)
- [“Process Summary report” on page 129](#)

DEDB Area Analysis reports

The DEDB Area Analysis reports include the following reports:

- Freespace Analysis report
- DB Record Profile Analysis report
- Segment Placement Analysis report

For more information about these reports, see the topic "DEDB Area Analysis reports" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

Segment Length Distribution report

This report shows the distribution of the length of variable segments for each segment type. It also contains information about the fixed-length segment that is compressed by the compression routine.

This report is optional and is generated for the area that is specified by the DEDBPC=(Y,SEGLDIST) keyword.

The distribution of SDEP length is not included in this report.

If no variable-length segment or compressed fixed-length segment is defined in the database, this report is not printed. In addition, if integrity verification for pointer fails, this report is not printed.

```
IMS HPFP UTILITIES - HPIC DEDBPC          "Segment Length Distribution Report"          PAGE:      5
5698-FPP V2R1                               2020-11-22 15:22:31

DBDNAME: DEDBJN23      AREANAME: DB23AR1
                        AREA NUMBER:      1
DISTRIBUTION OF SEGMENT LENGTHS
SEGMENTS REPORTED
  ROOTSEG1 DD1          DD11
SEGMENTS NOT REPORTED
- THE FOLLOWING SEGMENTS WERE NOT FOUND IN THE AREA
  no segment
- THE FOLLOWING SEGMENTS WERE DEFINED IN DBD AS FIXED-LENGTH SEGMENTS WITHOUT COMPRESSION ROUTINE
  DD12          DD2
```

Figure 89. Segment Length Distribution report (header part)

SEGMENTS REPORTED

Shows the names of the segments that are reported in the subsequent pages of the report.

SEGMENTS NOT REPORTED

Shows the names of the segments that are not reported in the subsequent pages of the report. These segments are not reported for either of the following reasons:

- THE FOLLOWING SEGMENTS WERE NOT FOUND IN THE AREA
- THE FOLLOWING SEGMENTS WERE DEFINED IN DBD AS FIXED-LENGTH SEGMENTS WITHOUT COMPRESSION ROUTINE

DBDNAME: DEDBJN23 AREANAME: DB23AR1
AREA NUMBER: 1

SC	SEGMENT NAME	SIZE RANGE	OCCURRENCES	PERCENTAGE	CUMULATIVE PERCENTAGE
1	ROOTSEG1	134 - 171	63	37.5 %	37.5 %
		172 - 209	0	0.0 %	37.5 %
		210 - 248	76	45.2 %	82.7 %
		249 - 286	2	1.2 %	83.9 %
		287 - 324	0	0.0 %	83.9 %
		325 - 363	0	0.0 %	83.9 %
		364 - 401	0	0.0 %	83.9 %
		402 - 439	0	0.0 %	83.9 %
		440 - 478	12	7.1 %	91.1 %
		479 - 516	3	1.8 %	92.9 %
		517 - 554	0	0.0 %	92.9 %
		555 - 593	0	0.0 %	92.9 %
		594 - 631	11	6.5 %	99.4 %
		632 - 669	0	0.0 %	99.4 %
		670 - 708	0	0.0 %	99.4 %
		709 - 746	0	0.0 %	99.4 %
		747 - 784	0	0.0 %	99.4 %
		785 - 823	0	0.0 %	99.4 %
		824 - 861	0	0.0 %	99.4 %
		862 - 900	1	0.6 %	100.0 %

TOTALS		134 - 900	168	100.0 %	
AVERAGE SEGMENT LENGTH=		243.2			
MORE THAN 90 PERCENT OF SEGMENT OCCURRENCES ARE INCLUDED IN THE RANGE 134 - 478.					

Figure 90. Segment Length Distribution report (main part)

SC

Segment code.

SEGMENT NAME

Segment name, as coded on the SEGM macro in the DBD.

SIZE RANGE

The size range (in bytes) of the part of the variable-length segment to be reported. For compressed segment, the length after compression is used for the size. The shortest and the longest sizes are actual values detected in the database, not extracted from the DBD. This report divides the size range into 20. If the segment length is distributed within a certain narrow range, the size range might become less than 20 or the size range might not be fixed.

OCCURRENCES

The number of occurrences that are included in this range.

PERCENTAGE

The percentage of occurrences of this range, and what percentage it makes of the total.

CUMULATIVE PERCENTAGE

The cumulative value of PERCENTAGE.

TOTALS

The minimum length, maximum length, and occurrences of the segment.

AVERAGE SEGMENT LENGTH

The average length of the data part in segments of this type in this partition or database.

MORE THAN 90 PERCENT OF SEGMENT OCCURRENCES ARE INCLUDED IN THE RANGE xxx - xxx.

This field shows the range that contains more than 90% of segment occurrences. This field is based on a more specific range that includes the average segment length.

Process Summary report

This report shows the summary of the process for each area.

DBDNAME	AREANAME	AREA NO	START DATE/TIME	END DATE/TIME	STATUS (If not blank, see messages in the message data sets.)
DEDBJN22	DB22AR2	3	2020-11-22 15:22:31	2020-11-22 15:22:32	
DEDBJN22	DB22AR0	1	2020-11-22 15:22:31	2020-11-22 15:22:32	
DEDBJN23	DB23AR2	2	2020-11-22 15:22:32	2020-11-22 15:22:32	
DEDBJN23	DB23AR3	3	2020-11-22 15:22:32	2020-11-22 15:22:32	
DEDBJN21	DB21AR0	1	2020-11-22 15:22:32	2020-11-22 15:22:32	

Figure 91. Process Summary report

DBDNAME

The name of DBD.

AREANAME

The name of the area.

AREA NO

The number of the area.

START DATE/TIME

The date and time when the HASH Check started for the area.

END DATE/TIME

The date and time when the HASH Check ended for the area.

STATUS

If HASH Check completed with no error, this column is blank. Otherwise, either of the following messages is printed in this column:

- HASH check did not complete. Errors in JCL or in HPIC process.
- HASH check completed. Pointer errors were detected.

SNAPDPIT DD data set

This data set contains the CI Map/CI Dump report generated by the HPFPU Hash Check support for IMS HP Image Copy.

For more information about CI Map/CI Dump report, see the topic "CI Map/CI Dump report" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

HFPAHST DD data set

This data set contains historical records. The HPFPU Hash Check support for IMS HP Image Copy adds one record for each DEDB area that it processes. The record contains the key space utilization and performance information for that area. This historical record is generated for the area that is specified by the DEDBPC=(Y,HISTORY) keyword.

If this feature is enabled, the HFPAHST data set must be preallocated with the following attributes:

- DSORG=PS
- RECFM=FB
- LRECL=100
- BLKSIZE=user-specified value

The DD statement in the JOB stream is specified as:

```
//HFPAHST DD DSN=HPFP.HISTORY,DISP=(MOD,KEEP,KEEP)
```

For the record layout of the historical file, see the topic "History file records layout" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

Important: To create historical records, the following APARs must be applied:

- APAR PH28270 to IBM IMS Fast Path Solution Pack for z/OS
- APAR PH30776 to IBM IMS High Performance Image Copy for z/OS

Examples: HPFPU Hash Check support for IMS HP Image Copy

The examples provided in the following topics show some of typical ways that you can use.

Subsections:

- [“Example 1: HPFPU Hash Check support for IMS HP Image Copy” on page 131](#)
- [“Example 2: HPFPU Hash Check and analysis of segment length distribution” on page 131](#)

Example 1: HPFPU Hash Check support for IMS HP Image Copy

This topic provides a JCL example to run an IMS HP Image Copy job, for a DEDB, with the HPFPU Hash Check support. This job takes image copies of two area data sets and generates a HASH evaluation report.

```
//HPICHASH EXEC PGM=FABJMAIN
//STEPLIB DD DISP=SHR,DSN=HPFP.SHFPLMD0
// DD DISP=SHR,DSN=HPS.SHPSLMD0
// DD DISP=SHR,DSN=IMSVS.SDFSRESL
//DFSRESLB DD DISP=SHR,DSN=IMSVS.SDFSRESL
//DFSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//ICEPRINT DD SYSOUT=*
//MSGOUT DD SYSOUT=*
//*
//REPORTS DD SYSOUT=*
//SNAPDPIT DD SYSOUT=*
//*
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//IMSDALIB DD DISP=SHR,DSN=IMSVS.MDALIB
//ICEIN DD *
GLOBAL DBRC=Y,
DEDBPC=Y,
ICHLQ=ICOUT.HFP,
UNIT=SYSDA,
SPACE=(CYL,100,100)
AIC DBD=DEDBJN23,AREA=DB23AR1
AIC DBD=DEDBJN23,AREA=DB23AR2
/*
```

Figure 92. HPFPU Hash Check support for IMS HP Image Copy

Example 2: HPFPU Hash Check and analysis of segment length distribution

This topic provides a JCL example to run an IMS HP Image Copy job, for a DEDB, with the HPFPU Hash Check support. This job takes image copies of two area data sets and generates, for each area, a HASH evaluation report, a Segment Length Distribution report, and a historical record.

```

//HPICHASH EXEC PGM=FABJMAIN
//STEPLIB DD DISP=SHR,DSN=HPFP.SHFPLMD0
// DD DISP=SHR,DSN=HPS.SHPSLMD0
// DD DISP=SHR,DSN=IMSVS.SDFSRESL
//DFSRESLB DD DISP=SHR,DSN=IMSVS.SDFSRESL
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//ICEPRINT DD SYSOUT=*
//DFSPRINT DD SYSOUT=*
//MSGOUT DD SYSOUT=*
//REPORTS DD SYSOUT=*
//SNAPDPIT DD SYSOUT=*
//HFP AHST DD DSN=&&SHST,DISP=(MOD,PASS,DELETE)
//*
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//IMSDALIB DD DISP=SHR,DSN=IMSVS.MDALIB
//ICEIN DD *
GLOBAL DBRC=Y,
DEDBPC=(Y,SEGLDIST,HISTORY),
ICHLQ=ICOUT.HFP,
UNIT=SYSDA,
SPACE=(CYL,100,100)
AIC DBD=DEDBJN23,AREA=DB23AR1
AIC DBD=DEDBJN23,AREA=DB23AR2
/*

```

Figure 93. HPFPU Hash Check and analysis of segment length distribution

DB Sensor support for IMS HP Image Copy

IMS HP Image Copy can call the DB Sensor support. The DB Sensor support is invoked during the image copy processing of an IMS HP Image Copy job. When the DB Sensor support is called in an IMS HP Image Copy job, DB Sensor collects statistics of DEDB areas and stores them as sensor data in the IMS Tools Knowledge Base Sensor Data repository.

Topics:

- [“Input for DB Sensor support for IMS HP Image Copy” on page 132](#)
- [“Output for DB Sensor support for IMS HP Image Copy” on page 132](#)
- [“Example: DB Sensor support for IMS HP Image Copy” on page 133](#)

Input for DB Sensor support for IMS HP Image Copy

To run an IMS HP Image Copy job for DEDB area data sets, you must specify additional DD statements and some keywords in the ICEIN control statements.

To activate the DB Sensor support, specify SENSOR=Y for the ICEIN control statement. When SENSOR=Y is specified, DB Sensor collects statistics from all the areas that are specified in the job.

The JCL requirements for using the DB Sensor support within an IMS HP Image Copy job are described in the *IMS High Performance Image Copy User's Guide*.

Output for DB Sensor support for IMS HP Image Copy

The outputs from the DB Sensor support are generated in the HFPSRT DD data set and the MSGOUT DD data set.

HFPSRT DD data set

This data set contains the Sensor Data Statistics report. For more information about Sensor Data Statistics report, see the topic "Sensor Data Statistics report" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

MSGOUT DD data set

This data set contains the messages that are issued by the DB Sensor support for IMS HP Image Copy. For more information, see the *IMS High Performance Image Copy User's Guide*.

Example: DB Sensor support for IMS HP Image Copy

This JCL example runs an IMS HP Image Copy job for a DEDB with the HPFPU Hash Check support and the DB Sensor support activated.

This job takes image copies of two area data sets and generates a HASH evaluation report. Sensor data is collected and stored in the Sensor Data repository of IMS Tools KB.

```
//HPICHASH EXEC PGM=FABJMAIN
//STEPLIB DD DISP=SHR,DSN=HPFP.SHFPLMD0
// DD DISP=SHR,DSN=HPS.SHPSLMD0
// DD DISP=SHR,DSN=IMSVS.SDFSRESL
// DD DISP=SHR,DSN=ITB.SHKTLOAD
//DFSRESLB DD DISP=SHR,DSN=IMSVS.SDFSRESL
//DFSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//ICEPRINT DD SYSOUT=*
//MSGOUT DD SYSOUT=*
//*
//REPORTS DD SYSOUT=*
//SNAPDPIT DD SYSOUT=*
//*
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//IMSDALIB DD DISP=SHR,DSN=IMSVS.MDALIB
//ICEIN DD *
GLOBAL DBRC=Y,
        DEBPC=Y,
        SENSOR=Y,
        ITKBSRVR=FPQSRV01,
        ICHLQ=ICOUT.HFP,
        UNIT=SYADA,
        SPACE=(CYL,100,100)
AIC DBD=DEDBJN23,AREA=DB23AR1
AIC DBD=DEDBJN23,AREA=DB23AR2
/*
```

Figure 94. DB Sensor support for IMS HP Image Copy

Chapter 11. Integration with IMS Database Recovery Facility

IMS Database Recovery Facility can call the HPFPU Hash Check and the FPA Build Index functions. These functions can be invoked within the IMS Database Recovery Facility job step.

Topics:

- “HPFPU Hash Check support for IMS Database Recovery Facility” on page 135
- “FPA Build Index support for IMS Database Recovery Facility” on page 136

HPFPU Hash Check support for IMS Database Recovery Facility

IMS Database Recovery Facility can call the HPFPU Hash Check function. The HPFPU Hash Check function is invoked within the IMS Database Recovery Facility job step.

Input for HPFPU Hash Check for IMS Database Recovery Facility

To invoke the HPFPU Hash Check function within the IMS Database Recovery Facility job step, you must specify additional DD statements and some keywords in the SYSIN control statements of IMS Database Recovery Facility.

The PC() keyword on the ADD command must be specified to run the HPFPU Hash Check function for the recovered DEDB area data sets. If you want to perform subset pointer checking during the hash check process, specify PC(SSPCHECK=YES). If you do not want the CI Map/CI Dump report to be printed, specify PC(FABASNAP=NO).

For primary address space JCL and subordinate address space JCL, see the *IMS Recovery Solution Pack IMS Database Recovery Facility User's Guide*.

Output for HPFPU Hash Check support for IMS Database Recovery Facility

The following topics describe the output produced by the HPFPU Hash Check function for IMS Database Recovery Facility.

For more information, see the *IMS Recovery Solution Pack IMS Database Recovery Facility User's Guide*.

FABAMSG DD data set

This data set contains messages issued by the HPFPU Hash Check function for IMS Database Recovery Facility.

FABARPRT DD data set

This data set contains the DEDB Area Analysis reports generated by the HPFPU Hash Check function for IMS Database Recovery Facility.

The DEDB Area Analysis reports include the following reports:

- Freespace Analysis report
- DB Record Profile Analysis report
- Segment Placement Analysis report

For more information about these reports, see the topic "DEDB Area Analysis reports" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

FABASNAP DD data set

This data set contains the CI Map/CI Dump report generated by the HPFPU Hash Check function for IMS Database Recovery Facility.

FPA Build Index support for IMS Database Recovery Facility

IMS Database Recovery Facility jobs can call the FPA Build Index function. The FPA Build Index function is invoked within the IMS Database Recovery Facility job step.

Input for FPA Build Index support for IMS Database Recovery Facility

To invoke the FPA Build Index support within the IMS Database Recovery Facility job step, you must specify additional DD statements and some keywords in the SYSIN control statements of IMS Database Recovery Facility.

The IB(BLD_SECONDARY()) keyword on the ADD command must be specified to enable the FPA Build Index support for the recovered DEDB area data sets.

For primary address space JCL and subordinate address space JCL, see the description in the *IMS Recovery Solution Pack IMS Database Recovery Facility User's Guide*.

Output for FPA Build Index support for IMS Database Recovery Facility

The HFPPRINT DD and HFPRPTS DD are the outputs produced by the FPA Build Index support for IMS Database Recovery Facility.

For information about these DD statements, see the topic "DD statements for the Build Index function" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

Chapter 12. Reference: Supplementary utility reports stored in IMS Tools KB

When `ITKBSRVR=servername` is specified, reports and messages of supplementary utilities can be stored in the output repository of IMS Tools KB.

Note: To store the reports and messages of HPFPU Hash Check support for IMS HP Image Copy in the IMS Tools KB Output repository, specify `ITKBSRVR=servername` in the ICEIN DD statement.

The following table summarizes the messages and reports generated by the supplementary utilities.

Table 16. Messages and reports stored in IMS Tools KB

Utility	Generated messages and reports	Whether the report can be stored in the IMS Tools KB Output repository
SDEP Space Utilization utility	SDEP Data Format-Message	-
	SDEP Utilization report	-
	SDEP History/Reports-Messages	-
Database Definition Record Create utility	FABCUR5-Messages report	-
	DBD Definition report	-
DEDB Reload Segment Data Set Create utility	FABCUR6-Messages report	-
	Audit Control report	-
DEDB Unload Segment Data Set Retrieve utility	FABCUR7-Messages report	-
	Audit Control report-Segment totals by input file	-
HD To DEDB Unload Data Set Conversion utility	FABCUR8-Messages report	-
	Audit Control report	-
DEDB Unload Conversion utility	Diagnostic Messages and Summary report	-
Standard format extract data interface module	Standard Format Extract Data Interface Output report	-
HPFPU Hash Check support for IMS HP Image Copy	DEDB Area Analysis report	Y
	Segment Length Distribution report	Y
	Process Summary report	Y
	SCAN DEDB AREA-Messages report	-
DB Sensor support for IMS HP Image Copy	Sensor Data Statistics report	Y
HPFPU Hash Check support for IMS Database Recovery Facility	DEDB Area Analysis report	-

Table 16. Messages and reports stored in IMS Tools KB (continued)

Utility	Generated messages and reports	Whether the report can be stored in the IMS Tools KB Output repository
FPA Build Index support for IMS Database Recovery Facility	Audit report	Y
	Processing report	Y
	DBD Definition report	Y
	Secondary Index Definition report	Y
	Secondary Index Processing report	Y

Chapter 13. Troubleshooting

These topics provide technical references to help you troubleshoot and diagnose IMS HP Fast Path Utilities supplementary utility problems.

Topics:

- [“Messages” on page 139](#)
- [“Gathering diagnostic information” on page 246](#)

Messages

Use the information in these topics to help you diagnose and solve supplementary utility problems.

For each message, the following accompanying information is provided where applicable:

Explanation:

This explains what the message text means, what caused the message to be issued, and what its variable entry fields are (if any).

System action:

This explains what the system will do next

User response:

This describes whether a response is necessary, what the appropriate response should be, and what the resulting effect is on the system or program.

Message prefixes

The following table shows the prefixes of messages and the utility or the process that issues the messages.

Table 17. Message prefixes	
Prefix	Utility or process
FABA	SDEP Space Utilization utility and HPFPU Hash Check process
FABC	Supplementary utilities
FABD	Common routines of supplementary utilities
FABU	Diagnostics Aid utilities. For Diagnostics Aid messages, see the <i>IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide</i> .
HFPB	FPA Build Index support for IMS Database Recovery Facility

Message suffixes

Some messages provide additional information by including the following suffixes:

A

Indicates that operator intervention is required before processing can continue.

I

Indicates that the message is informational only.

W

Indicates that the message is a warning to alert you to a possible error condition.

E

Indicates that an error occurred, which might or might not require operator intervention.

FABA messages

The following information is about messages and codes that begin with FABA.

FABA4000I **DBDNAME:** *dbdname* **AREA:**
areaname
- DEDB POINTER CHECKER ENDED
NORMALLY

Explanation

DEDB Pointer Checker detected no pointer errors during processing.

System action

DEDB Pointer Checker terminates normally.

FABA4001I **DBDNAME:** *dbdname* **AREA:**
areaname
- PROCESSING COMPLETE

Explanation

This message is generated when all data for the area *areaname* of the database *dbdname* has been processed.

System action

Processing continues.

FABA4002I **DBDNAME:** *dbdname* **AREA:**
areaname
- SUBSET POINTER CHECK
PROCESSING

Explanation

The subset pointer check is processed because '#' was specified on column 49 of the SYSIN DD.

System action

Processing continues.

FABA4003I **DBDNAME:** *dbdname* **AREA:**
areaname
- DATASPACE REQUESTED FOR
SSPTR CHECK PROCESS

Explanation

The subset pointer check is processed by using the data space.

System action

Processing continues.

FABA4004I **DBDNAME:** *dbdname* **AREA:**
areaname
- SSPTR CHECK PROCESS
REQUESTED, BUT SSPTR WAS NOT
DEFINED

Explanation

The subset pointer check was requested, but no subset pointer was defined in the processed database.

System action

Processing continues.

FABA4005I **DBDNAME:** *dbdname* **AREA:**
areaname
- SDEP PROCESSING STARTED
- LB: *cycle# rel-byte-addr*
- LE: *cycle# rel-byte-addr*

Explanation

This message is generated when SDEP processing is about to start. LB specifies the logical beginning of the SDEP part of the area, and LE specifies the logical end of the SDEP part of the area.

System action

Processing continues.

FABA4006I **DBDNAME:** *dbdname* **AREA:**
areaname
- CI MAP/DUMP FOR CI: *xxxxxxx*
GENERATED. DUMP NO.=*nnn*

Explanation

This is an informational message to let you know that the CI map or the CI dump for CI *xxxxxxx* is generated in a data set that is specified with the SNAPDPIT DD statement for ICE hash check or with the FABASAP DD statement for IMS DRF hash check. A dump number *nnn* is assigned.

System action

See the user response section.

User response

For both system action and user response, see the message issued immediately before this message to determine the error that caused this dump to be generated.

FABA4007I **DBDNAME: *dbdname* AREA: *areaname***
- NO SDEP SEGMENTS FOUND
- SDEP PROCESSING BYPASSED

Explanation

'TYPE=SEQ' was specified in the DBD, but DEDB Pointer Checker determined that the SDEP part associated with the area *areaname* was empty. This condition is detected if the 'logical beginning' and the 'logical end' contained in DMAC have the same value, or if their values differ by 4.

System action

Processing continues. The SDEP processing is bypassed.

FABA4011W **DBDNAME: *dbdname* AREA: *areaname***
- POINTER ERRORS DETECTED
DURING HASH CHECK

Explanation

DEDB Pointer Checker detected pointer errors during processing.

System action

DEDB Pointer Checker terminates normally.

User response

Determine the cause of the error, using the other messages generated. Correct the problem and rerun the job, or run the full pointer checker job. For more information, see the following topics in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*:

- "DEDB integrity verification"
- "Running the DEDB Pointer Checker process"

FABA4012W **DBDNAME: *dbdname* AREA: *areaname***
- CI MAP/DUMP FUNCTION IS
IGNORED. REASON CODE *nn*

Explanation

The CI map or the CI dump function is ignored, for one of the following reasons:

- Reason code 01
SNAPDPIT data set for ICE hash check was not provided, the FABASNAP data set for IMS DRF hash check was not provided, open failed, or I/O failed.

- Reason code 06
One hundred CI maps or dumps, the maximum number, have already been generated.
- Reason code 07
GETMAIN failed to obtain space for internal blocks.
- Reason code 08
All entries in the block map table (internal table) have been consumed.

System action

DEDB Pointer Checker bypasses the CI map or CI dump function and continues processing.

User response

If a CI map or a CI dump is needed, correct the errors and rerun the job.

FABA4013W **DBDNAME: *dbdname* AREA: *areaname***
- SPACE MAP AT RBA: *xxxxxxxx*
OFFSET: *zzz* HAS CONTROL WORD
DISCREPANCY

Explanation

The CI corresponding to space map at RBA *xxxxxxxx* OFFSET *yyyy* should be a first allocatable CI. (That is, 1 byte from OFFSET *yyyy* should be x'80'.) But the space map shows that the CI is already allocated. (That is, 1 byte from OFFSET *yyyy* is x'40'.)

System action

DEDBPC sets an end-of-job return code of 2, and continues processing.

User response

Correct the error and rerun the job.

FABA4015W **DBDNAME: *dbdname* AREA: *areaname***
- UOW# DISCREPANCY: CI AT
RBA: *xxxxxxxx* IOVF SPACE
MAP: *xxxxxxxx* IOVF CI PREFIX:
xxxxxxxx

Explanation

The UOW number in space map that corresponds to CI at RBA *xxxxxxxx* and the number in IOVF CI prefix at RBA *xxxxxxxx* are different.

System action

DEDBPC sets an end-of-job return code of 2, and continues processing by using the value of the space map.

User response

Correct the error and rerun the job.

FABA4022E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - INVALID BLK TYPE ID IN CI AT RBA: <i>xxxxxxx</i> - CI BYPASSED (DATA VALUE: <i>yy</i> OFFSET: <i>zzz</i>)
------------------	--

Explanation

During the serial 'deblocking' of the CI at RBA *xxxxxxx*, DEDB Pointer Checker encountered an incorrect block type (DBLKBTID). Or the problem may be that a segment or an FSE in the specified CI has an incorrect length.

System action

DEDB Pointer Checker bypasses the CI in error, and continues processing.

User response

Correct the errors, and rerun the job.

FABA4023E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - INVALID SEGM CODE IN CI AT RBA: <i>xxxxxxx</i> - CI BYPASSED (DATA VALUE: <i>yy</i> OFFSET: <i>zzz</i>)
------------------	--

Explanation

During the serial "deblocking" of the CI at RBA *xxxxxxx*, DEDB Pointer Checker encountered an incorrect segment code. Or the problem may be that a segment or an FSE in the specified CI has an incorrect length.

System action

DEDB Pointer Checker bypasses the CI in error, and continues processing.

User response

Correct the error, and rerun the job.

FABA4024E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - TOTAL FSE LENGTH - RBA: <i>xxxxxxx</i>
------------------	---

Explanation

DEDB Pointer Checker determined that the total free space in the CI at RBA *xxxxxxx* as calculated by 'chasing' the FSE chain did not correspond to the value calculated during the serial 'deblocking' of that CI. This condition might have been caused by an incorrect FSE chain, or by an incorrect FSE or segment length.

System action

DEDB Pointer Checker bypasses the CI in error and continues processing.

User response

For a description of the corrective action required, see the topic "DEDB integrity verification" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

FABA4025E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - TOTAL NO. OF FSE - RBA: <i>xxxxxxx</i>
------------------	---

Explanation

DEDB Pointer Checker determined that the number of free space elements in the CI at RBA *xxxxxxx* as calculated by 'chasing' the FSE chain did not match the value calculated during the serial 'deblocking' of that CI. This condition might have been caused by an incorrect FSE chain or by an incorrect FSE or segment length.

System action

DEDB Pointer Checker bypasses the CI in error, and continues processing.

User response

For a description of the corrective action required, see the topic "DEDB integrity verification" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

FABA4026E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - CI "SPACE USAGE" - RBA: <i>xxxxxxx</i>
------------------	---

Explanation

DEDB Pointer Checker determined that the sum of the free space element, scrap, and segment lengths in the CI at RBA xxxxxxxx encountered during the serial 'deblocking' of that CI, was not equal to the usable space of the CI. This condition might have been caused by an incorrect FSE chain, or by an incorrect FSE or segment length.

System action

DEDB Pointer Checker bypasses the CI in error and continues processing.

User response

For a description of the corrective action required, see the topic "DEDB integrity verification" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

FABA4027E **DBDNAME: dbdname AREA: areaname**
- FSE CHAIN POINTS TO A NON FSE
- RBA: xxxxxxxx

Explanation

While following the FSE chain for the CI at RBA xxxxxxxx (that is, a first byte that was neither X'80' nor X'70'), DEDB Pointer Checker encountered an incorrect free space element.

System action

DEDB Pointer Checker bypasses the CI in error and continues processing.

User response

For a description of the corrective action required, see the topic "DEDB integrity verification" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

FABA4028E **DBDNAME: dbdname AREA: areaname**
- SEGMENT AT RBA: xxxxxxxx (SEGCD: yy) HAS PCF/PCL/SSPTR DISCREPANCY

Explanation

While checking the PCF/PCL/SSPTR pointer inter-dependencies, DEDB Pointer Checker encountered an error in the segment at RBA xxxxxxxx.

System action

Processing continues.

User response

Make sure that the pointer values contained in the segment at RBA xxxxxxxx meet the following criteria:

- If the PCF pointer value is zero, the associated PCL pointer value and subset pointer value must also be zero.
- If the PCF pointer value is nonzero, the associated PCL pointer value must also be nonzero.

For a description of the corrective action required, see the topic "DEDB integrity verification" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*.

FABA4029E **DBDNAME: dbdname AREA: areaname**
- POINTER (PTF/PCF) ERRORS

Explanation

DEDB Pointer Checker performed a checksum validation test of the RBAs of all segments in the specified area versus the values of their PCF and PTF pointers. The test failed. This shows that the area *areaname* contains PTF/PCF pointer integrity problems.

System action

Processing continues.

User response

Run the full pointer checker job with the 'TYPRUN=PTRALL' mode, if required. For more information, see the following topics in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*:

- "DEDB integrity verification"
- "Running the DEDB Pointer Checker process"

FABA4030E **DBDNAME: dbdname AREA: areaname**
- POINTER (PCL) ERRORS

Explanation

DEDB Pointer Checker performed a checksum validation test of the RBAs of all segments referred to by the PCL pointer versus the values of the PCL pointers. The test failed. This shows that the area *areaname* contains PCL pointer integrity problems.

System action

Processing continues.

User response

Run the full pointer checker job with the 'TYPRUN=PTRALL' mode if required. For more information, see the following topics in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*:

- "DEDB integrity verification"
- "Running the DEDB Pointer Checker process"

FABA4031E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - POINTER (SDEP) ERRORS
------------------	--

Explanation

DEDB Pointer Checker performed a checksum validation test of the RBAs of all ROOT and SDEP segments in the specified area versus the values of their SDEP pointers. The test failed. This shows that the area *areaname* contains SDEP pointer integrity problems.

System action

Processing continues.

User response

If the exact RBA of the errors is needed, run the full pointer checker job with the 'TYPRUN=PTRALL' mode. For more information, see the following topics in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*:

- "DEDB integrity verification"
- "Running the DEDB Pointer Checker process"

FABA4032E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - SEGMENT AT RBA: <i>xxxxxxxx</i> (SEGCD: <i>yy</i>) HAS INVALID SDEP POINTER VALUE
------------------	--

Explanation

The SDEP pointer value contained in the segment at RBA *xxxxxxxx* does not point the SDEP part of the DEDB area.

System action

Processing continues.

User response

Run the full pointer checker job from program FABADA1 to FABADA5 to determine the cause of the error. For more information, see the following topics in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*:

- "DEDB integrity verification"
- "Running the DEDB Pointer Checker process"

FABA4033E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - INVALID VALUE IN "DMACXVAL"/"DMACNXTS" - SDEP PROCESSING BYPASSED
------------------	--

Explanation

DEDB Pointer Checker encountered an error while validating the 'logical beginning' and the 'logical end' of the SDEP part associated with the specified area. The 8-byte field of 'DMACXVAL' contains a value higher than the 8-byte field of 'DMACNXTS.'

System action

DEDB Pointer Checker continues processing. The SDEP processing is bypassed.

User response

Correct the error and rerun the job, or run the full pointer checker job to determine the cause of the error.

FABA4034E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - CI AT RBA: <i>xxxxxxxx</i> CONTAINS NEGATIVE SEGMENT/FSE LENGTH - CI BYPASSED
------------------	--

Explanation

DEDB Pointer Checker encountered a segment or an FSE with a negative length during the serial 'deblocking' of the CI at RBA *xxxxxxxx*.

System action

DEDB Pointer Checker bypasses the CI in error, and continues processing.

User response

Correct any errors and rerun the job. If this situation persists, report it to the system operation personnel.

FABA4035E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i>
------------------	--

- CI AT RBA: xxxxxxxx CONTAINS INVALID SEGMENT/FSE LENGTH
- CI BYPASSED

Explanation

DEDB Pointer Checker encountered a segment or an FSE with an incorrect length during the serial 'deblocking' of the CI at RBA xxxxxxxx.

System action

DEDB Pointer Checker bypasses the CI in error, and continues processing.

User response

Correct any errors and rerun the job. If this situation persists, report it to the system operation personnel.

FABA4036E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - INCONSISTENT "DBDNAME": CONTROL CARD: <i>dbdname1</i> DMAC CI: <i>dbdname2</i>
------------------	---

Explanation

The dbdname specified on the control statement does not match the dbdname in the DMAC CI.

System action

Processing continues. This area processing is bypassed.

User response

Make sure that the dbdname on the control statement is correct for the database. Correct the error and rerun the job.

FABA4037E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - INCONSISTENT "DEDB AREA DDNAME": CONTROL CARD: <i>areaname1</i> DMAC CI: <i>areaname2</i>
------------------	--

Explanation

The database ddname specified on the control statement does not match the database ddname in the DMAC CI.

System action

Processing continues. This area processing is bypassed.

User response

Make sure that the database ddname on the control statement is correct for the database. Correct the error and rerun the job.

FABA4038E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - DEDB INCONSISTENCY: FIELD: <i>fieldname</i> VALUE IN DBDLIB: <i>value-1</i> VALUE IN DMAC: <i>value-2</i>
------------------	--

Explanation

DEDB Pointer Checker determined that the specifications found in the DBD did not match those found in the DMAC.

System action

Processing continues. This area processing is bypassed.

User response

Make sure that the DBD is correct for the database. Correct the error and rerun the job.

FABA4039E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - MORE THAN 10 BAD CI'S ENCOUNTERED
------------------	--

Explanation

DEDB Pointer Checker encountered more than 10 CIs that contained one or more incorrect data items.

System action

Processing continues. This area processing is bypassed.

User response

See messages FABA4022E, FABA4023E, FABA4024E, FABA4025E, FABA4026E, and FABA4027E to find the CIs in error. Correct any errors and rerun the job. If this situation persists, report it to the system operation personnel.

FABA4040E	DEVTYPE FAILED FOR DDNAME HFPAST
------------------	---

Explanation

The HPFPU Hash Check support for IMS HP Image Copy issued an SVC 24 (DEVTYPE) to obtain

information about the HFP AHST DD data set, but the attempt failed.

System action

Processing ends with a return code of 16.

User response

Correct any errors, and rerun the job. If this situation persists, report it to systems operations personnel.

FABA4041E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - SSPTR TARGET SC IS INVALID (SSPTR: xxxxxxxx TARGET: xxxxxxxx)
------------------	--

Explanation

The segment pointed by the subset pointer has an unexpected segment type.

System action

Processing continues.

User response

Repair the database, and rerun the job.

FABA4042E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - SSPTR TARGET SEGMENT NOT FOUND (SSPTR: xxxxxxxx TARGET: xxxxxxxx)
------------------	--

Explanation

No valid segment type is placed where the subset pointer points.

System action

Processing continues.

User response

Repair the database and rerun the job.

FABA4043E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - SSPTR TARGET RBA IS INVALID (SSPTR: xxxxxxxx TARGET: xxxxxxxx)
------------------	---

Explanation

The place where the subset pointer points does not contain any segment data.

System action

Processing continues.

User response

Repair the database and rerun the job.

FABA4044E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - DSPSERV <i>func-cd</i> FAILED (RC=xxxx,RSN=yyyyyyyy)
------------------	---

Explanation

The DSPSERV macro failed. The function code of the DSPSERV macro is shown in *func-cd* and the return code and reason code are shown in *xxxx* and *yyyyyyyy*, respectively.

System action

DEDB Pointer Checker terminates with an abend code of 4044.

User response

For further explanation of the error, see the *MVS™ Programming: Authorized Assembler Services Reference*. Correct any errors and rerun the job. Or, do not use the data space when rerunning the job.

FABA4045E	DBDNAME: <i>dbdname</i> AREA: <i>areaname</i> - ALESERV <i>func-cd</i> FAILURE OCCURRED (RC=xxxx)
------------------	--

Explanation

The ALESERV macro failed. The function code of the DSPSERV macro is shown in *func-cd* and the return code is shown in *xxxx*.

System action

DEDB Pointer Checker terminates with an abend code of 4045.

User response

For further explanation of the error, see the *MVS Programming: Authorized Assembler Services Reference*. Correct any errors and rerun the job. Or, do not use the data space when rerunning the job.

FABA4046E **DBDNAME: *dbdname* AREA: *areaname***
- GETMAIN FAILURE OCCURRED

Explanation

DEDB Pointer Checker issued a GETMAIN macro. The return code signifies that the attempt to do so was unsuccessful.

System action

DEDB Pointer Checker terminates with an abend code of 4046.

User response

Increase the region size parameter on the EXEC statement, and rerun the job.

FABA4047E **OPEN FAILURE OCCURRED**
(DD:*ddname*)

Explanation

OPEN processing failed for the file associated with the DD statement specified.

System action

DEDB Pointer Checker terminates with an abend code of 4047.

User response

Make sure that a DD statement is present for the *ddname* specified, and that it is specified correctly. Correct any errors, and rerun the job.

FABA4048E **LOAD FAILURE OCCURRED**
(*dbdname*)

Explanation

DEDB Pointer Checker issued a LOAD macro to load the specified member *dbdname* from the DBD library, but the LOAD processing failed.

System action

DEDB Pointer Checker terminates with an abend code of 4048.

User response

Make sure that the specified member *dbdname* exists in the DBD library. Correct the error, and rerun the job.

FABA4049E **DBDNAME: *dbdname* AREA: *areaname***
- FREEMAIN FAILURE OCCURRED

Explanation

An internal error occurred in DEDB Pointer Checker.

System action

DEDB Pointer Checker terminates with an abend code of 4049.

User response

Correct any obvious errors and rerun the job. If the problem persists, save the entire run listing (including the dump, JCL, and FPB version reports) and call IBM for support.

FABA4050E **DBDNAME: *dbdname* AREA: *areaname***
- SORT FOR SSPTR CHECK FAILED

Explanation

DEDB Pointer Checker linked the DFSORT program internally to sort records in the data set associated with the SORTIN DD statement, and DFSORT returned an error status.

System action

DEDB Pointer Checker terminates with an abend code of 4050.

User response

Make sure that SYSOUT, SORTIN, SORTOUT, SORTWK01, SORTWK02, and SORTWK03 DD statements are correctly specified. If there are DFSORT messages on the data set associated with the SYSOUT DD statement, check those messages and correct any errors.

FABA4051E **MESSAGE TEXT NOT FOUND**
message-id

Explanation

DEDB Pointer Checker attempted to print an error message but could not find it in the message table in module FABAMSGS.

System action

DEDB Pointer Checker terminates with an abend code of 4051.

User response

Verify that FPB, including all current maintenance, was installed correctly. Reinstall FPB (including all maintenance), if necessary, and rerun the job. If the problem persists, save the entire run listing (including the dump, JCL, and FPB version reports) and call IBM for support.

FABA4052E INVALID MESSAGE FLAG

Explanation

DEDB Pointer Checker attempted to print an error message, but an incorrect flag was supplied to module FABAMSGS.

System action

DEDB Pointer Checker terminates with an abend code of 4052.

User response

Verify that FPB, including all current maintenance, was installed correctly. Reinstall FPB (including all maintenance), if necessary, and rerun the job. If the problem persists, save the entire run listing (including the dump, JCL, and FPB version reports) and call IBM for support.

**FABA4053E DBDNAME: *dbdname* AREA:
areaname
- INVALID UOW NUMBER
DETECTED ON IOVF CI AT RBA:
xxxxxxx
- CI BYPASSED**

Explanation

DEDB Pointer Checker encountered an incorrect UOW number in IOVF CI at RBA xxxxxxxx.

System action

Processing continues.

User response

Correct any errors and rerun the job.

**FABA4055E DBDNAME: *dbdname* AREA:
areaname
- CURRENT AREA NOT FOUND**

Explanation

The area cannot be found in DBD. This message is issued only by DEDB hash pointer checking invoked under IMS DRF.

System action

DEDB Pointer Checker ends with an abend code of 4055.

User response

Specify the correct area name in the control statement, and resubmit the job.

**FABA4056E LOAD FAILED FOR *dbdname*
- DBD NOT FOUND
- DBD SIZE IS ZERO
- GETMAIN FAILED
- LOAD MACRO ERROR**

Explanation

DBD cannot be loaded. This message is issued only by DEDB hash pointer checking invoked under IMS DRF.

System action

DEDB Pointer Checker ends with an abend code of 4056.

User response

Correct the error, and rerun the job. If this situation persists, contact IBM Software Support.

**FABA4057I DBDNAME: *dbdname* AREA:
areaname - DELETED SDEP
SEGMENT(S) FOUND BETWEEN
SDEP LB AND LE**

Explanation

This message is generated when a deleted SDEP segment between SDEP LB and LE was first found during HASH checking.

System action

Processing continues.

User response

None. This message is informational.

**FABA4058E HPIC VERSION CHECK MODULE
ENDED WITH AN ERROR. RETURN
CODE=*nn*.**

Explanation

The image copy version check process failed with an error.

System action

DEDB Pointer Checker ends with an abend code of 4058.

User response

Contact IBM Software Support.

**FABA4059W DBDNAME: *dbdname* AREA:
 areaname
 - DMAC INCONSISTENCY
 DETECTED DURING HASH CHECK**

Explanation

HPFPU Hash Check support for IMS HP Image Copy detected that the specifications found in the DBD did not match those found in the DMAC. Message FABA4038E provides details of the inconsistency found.

System action

Processing continues. Hash check processing for this area is bypassed.

User response

Ensure that the DBD is correct for the database. Correct the error and rerun the job.

FABA4093E INTERNAL ERROR OCCURRED.

Explanation

DEDB Pointer Checker detected an internal error.

FABC messages

The following information is about messages and codes that begin with FABC.

FABC0100I FABCUR1 ENDED NORMALLY

Explanation

This message is generated when all requested processing has been completed without errors.

System action

Program FABCUR1 ends with a return code of 0.

System action

DEDB Pointer Checker ends with an abend code of 4093.

User response

Contact IBM Software Support.

FABA4095E RECON ACCESS FAILED. *subtext*

Explanation

An error was detected in the RECON access processing. One of the following subtexts is issued:

- DBRC LIST COMMAND IS NOT COMPLETED.
 RC=xxxxxxx
- SYSPRINT DD FOR DBRC LIST COMMAND IS SPECIFIED AS DUMMY
- INTERNAL ERROR OCCURRED
- FUNC=ffffffff RETURN CODE=xxxxxxx REASON
 CODE=xxxxxxx
 KEYS: DBD=*dbdname* DDN=*ddname*
 KEYTYPE=xxxxxxxxxxx

System action

Program FABADA1 ends with an abend code of 4095.

User response

Check the DBRC message preceding this message. Follow the response in that message, and rerun the job.

User response

None. This message is informational.

**FABC0100W FABCUR1 ENDED WITH
 WARNINGS**

Explanation

This message is generated when trivial error conditions were encountered by program FABCUR1.

System action

FABCUR1 ends with a return code of 4.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR1. Correct the problem and rerun the job, or continue with reload processing, as desired.

FAB00100E FABCUR1 ENDED WITH ERRORS

Explanation

This message is generated when nontrivial error conditions were encountered by program FABCUR1.

System action

FABCUR1 ends with a return code of 8.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR1. Correct the problem, and rerun the job.

**FAB00101I DATA SET UNLOADED FOR AREA
zzzzz (AREA NAME: areaname)
- DD NAME: ddname DS NAME:
dsname**

Explanation

This message is generated when program FABCUR1 selects the area data set specified for unloading the area.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

**FAB00105I PROCESSING COMMENCES FOR
AREA zzzzz (DDNAME: ddname)
[(BUFND = zzz) | (ACCESS = FAST)]**

Explanation

This message is generated when program FABCUR1 dispatches an unload subtask to process the specified area. BUFND = parameter is the value used when the area is opened. If ACCESS = FAST was specified, ACCESS = FAST is displayed instead of BUFND = parameter value.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

**FAB00106I PROCESSING COMPLETED FOR
AREA zzzzz (DDNAME: ddname)
[AREA IS EMPTY | (n1 / n2)]**

Explanation

This message is issued when an unload subtask notifies program FABCUR1 that the specified area has been successfully unloaded. If the area is empty, the message text that specifies this is issued. n1 is the user-task ID. n2 is the number of times that FABCUR1 is put into a wait state while the database segment records are written.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

**FAB00107W NO SEGMENTS WILL BE
RELOADED TO AREA zzzzz
(AREaname: areaname)**

Explanation

No segment will be reloaded to the specified area.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

None.

**FAB00108W UNLOADED DATA FOR AREA zzzzz
(AREA NAME: areaname) MAY BE
WRONG
- DUE TO reason**

Explanation

When unload operation completed for the area specified, DBRC gives one of the following reasons that implies that the unloaded data may be wrong.

- RECOVERY NEEDED STATUS: By the IMS online system or by the DBRC batch command the area was

made recovery needed during unload processing.
Existence of EEQE: An EEQE has been created by the IMS online system during unload processing.

- CONCURRENT UPDATE: The IMS online system accessed the area with update intent during unload processing.
- AREA DROPPING FROM DBRC: The area registration was dropped by the DBRC batch command from DBRC during unload processing.
- ADS UNAVAILABLE STATUS: The ads were made unavailable during unload processing by the IMS online system or by the DBRC batch command.
- ADS DROPPING FROM DBRC: During unload processing, the ads registration was dropped by the DBRC batch command from DBRC.

System action

Program FABCUR1 sets an end-of-job return code 4 and continues processing.

User response

Make sure that there was no update operation by the IMS online system during unload operation. If there is a possibility that the area was updated during unload processing, then rerun the job for the area specified.

FABC0109I	AREA zzzzz (AREaname: areaname) IS NOT REGISTERED IN DBRC
------------------	--

Explanation

Program FABCUR1 found that the specified area was not registered in DBRC.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

FABC0110W	NO RECORDS WRITTEN TO DDNAME DURDzzzO / XDzzzzzO
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Attempt to determine if there should have been any segment data records written to the specified output file. Verify that the DD statement NEWACB correctly identifies the proper data set, and that the DBDGEN and ACBGEN for the database being processed were performed correctly. Check that the randomizer module is specified correctly. Review the FILECTL specifications, if any. Correct the problem and rerun the job, or continue with reload processing, as desired.

FABC0111I	DBRC=Y IS SPECIFIED
------------------	----------------------------

Explanation

DBRC=Y is specified in the EXEC parameter of the FABCUR1 JCL. Program FABCUR1 will establish DBRC interface and obtain area information from DBRC.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

FABC0112I	DBRC=N IS SPECIFIED - EEQE DETECTION NOT PERFORMED
------------------	---

Explanation

DBRC=N is specified in the EXEC parameter of the FABCUR1 JCL. Program FABCUR1 does not establish DBRC interface and does not check the existence of EEQEs.

System action

FABCUR1 continues processing.

User response

If the area is registered in DBRC, get DBRC RECON list and make sure there are no EEQEs registered in DBRC for the area. If there is an EEQE for the area, recover the area and rerun the job.

FABC0113I	AREA zzzzz (AREaname: areaname) DDNAME: ddname IS UNAVAILABLE IN DBRC
------------------	--

Explanation

Program FABCUR1 found that the specified area data set was unavailable in DBRC.

System action

FABCUR1 ignores the area data set and continues processing.

User response

Get a LIST.RECON output report, and specify an unused area data set name. Then, specify the name in the DARVSAM DD statement, and rerun the job.

FAB00114I	AREA zzzzz (AREANAME: areaname) DDNAME: ddname NOT SAME DSNAME BETWEEN DD STATEMENT AND DBRC
------------------	---

Explanation

Program FABCUR1 found that the area data set dsname specified in ddname DD statement was not same one registered in DBRC.

System action

FABCUR1 ignores the area data set and continues processing.

User response

Get a LIST.RECON output report, and specify an unused area data set name. Then, specify the name in the DARVSAM DD statement, and rerun the job.

FAB00115I	SCHEDULING PARAMETERS: NO. UTASK'S : zz9 UOW BFR SIZE: z,zzz,zz9 BYTES
------------------	---

Explanation

This message is generated, when the STATS keyword is specified on the DBDNAME control statement, to detail the parameters being used to dispatch and manage the unload subtasks.

System action

Program FABCUR1 continues processing.

User response

None. This message is informational.

FAB00116W	UTASK zzz TERMINATED DUE TO STORAGE CONSTRAINTS
------------------	--

Explanation

Program FABCUR1 issued an OPEN for the ACBs associated with the next area to be processed, when

preparing to dispatch an unload subtask to unload it. The return code from VSAM specified that the request failed because of insufficient storage being available for the required buffers and control blocks.

System action

FABCUR1 sets an end-of-job return code of 4, dispatches the specified unload subtask with a control code indicating that it should terminate itself, and continues processing.

User response

Review the unload region-size calculations (especially if BUFND overrides are being used on the area data sets). Check that the REGION= parameter is coded correctly on the EXEC statement for FABCUR1.

FAB00117I	EXIT ROUTINE <i>exitname</i> "END" CALL FINISHED -first 80 bytes characters of the message that user exit routine returned -subsequent 48 bytes characters of the message that user exit routine returned
------------------	--

Explanation

Program FABCUR1 called the user exit routine *exitname* with "END" call and the exit routine returned the message specified.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

FAB00120I	CARD xx: zzzz...zzzz
------------------	-----------------------------

Explanation

This message is generated to show the control statement currently being processed.

System action

Program FABCUR1 continues processing.

User response

None. This message is informational.

FAB00121W	ERROR DETECTED NEAR COLUMN xx
------------------	--------------------------------------

Explanation

Program FABCUR1 detected an error in the control statement currently being processed. (See the immediately preceding FABC0120I message.)

System action

FABCUR1 continues processing, and issues one or more other FABC01xx messages.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR1. Correct the problem and rerun the job, or continue with reload processing, as desired.

FABC0121E	ERROR DETECTED NEAR COLUMN xx
------------------	--

Explanation

See message number FABC0121W.

System action

See message number FABC0121W.

User response

See message number FABC0121W.

FABC0122W	BLANK/INVALID CONTROL CARD
------------------	-----------------------------------

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an end-of-job return code of 4, and continues processing.

User response

Remove the specified control statement in subsequent executions of FABCUR1.

FABC0123E	UNKNOWN KEYWORD
------------------	------------------------

Explanation

Program FABCUR1 encountered a control statement with a value starting in column one that is not one of the valid control statements.

System action

FABCUR1 ends with an abend code of 3728.

User response

Correct, or remove, the specified control statement.

FABC0123W	UNKNOWN KEYWORD
------------------	------------------------

Explanation

Program FABCUR1 encountered a control statement with a value starting in column one that is not one of the valid control statement types.

System action

FABCUR1 ends with an abend code of 3728.

User response

Correct, or remove, the specified control statement. Rerun the job.

FABC0125E	1ST CONTROL CARD NOT DBDNAME= CARD
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 ends with an abend code of 3728.

User response

The control statement stream must include one DBDNAME control statement, and it must be the first statement in the stream. Correct the control statement stream. Rerun the job.

FABC0126E	MULTIPLE DBDNAME= CARDS ENCOUNTERED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 ends with an abend code of 3728.

User response

The control statement stream must include only one DBDNAME control statement, and it must be the first statement in the stream. Correct the control statement stream. Rerun the job.

FABC0127I **- FOLLOWING VALUES ARE
DEFINED BY SITE DEFAULT TABLE
(xxxxxxx)**
- **keyword=value**
- **keyword=value**

Explanation

This message is generated to show the site default table (FABCOP1D/FABCOP3D/FABCOP6D/FABCOP9D) being processed.

System action

Program FABCUR1/FABCUR3/FABCUR6/FABCUR9 uses the values as the default values for the control statement, and continues processing.

User response

None. This message is informational.

FABC0130E **INVALID DBDNAME= CONTROL
CARD
SYNTAX ERROR DETECTED**

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the DBDNAME control statement. Correct the errors, and rerun the job.

FABC0131E **INVALID DBDNAME= CONTROL
CARD
DBDNAME MISSING/INVALID**

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the DBDNAME control statement. Correct the error. Rerun the job.

FABC0133E **INVALID DBDNAME= CONTROL
CARD
"REORG" AND "HIERCHNG"/
"RMODTYPE"/"NEWDBDNM"
KEYWORDS ARE MUTUALLY
EXCLUSIVE**

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the DBDNAME control statement. Correct the error. Rerun the job.

FABC0135E **INVALID AREACTL= CONTROL
CARD
AREA NO(S) SPECIFICATION
MISSING/INVALID**

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the AREACTL control statement. Correct the error. Rerun the job.

FABC0136W **INVALID AREACTL= CONTROL
CARD
AREA xxxxx PREVIOUSLY
SPECIFIED**

Explanation

Self-explanatory. xxxxx is the area number.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Remove the duplicate specification for the specified area in subsequent executions of FABCUR1.

FABC0137W	INVALID AREACTL= CONTROL CARD HIERCHNG=YES SPECIFIED; AREACTL=ALL REQUIRED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Specify AREACTL=ALL or remove HIERCHNG=YES specification on the control statement in subsequent executions of FABCUR1.

FABC0140E	INVALID FILECTL= CONTROL CARD FILE NO(S) SPECIFICATION MISSING/INVALID
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the FILECTL control statement. Correct the control statement stream, and rerun the job.

FABC0141E	INVALID FILECTL= CONTROL CARD AREA zzzzz PREVIOUSLY SPECIFIED
------------------	--

Explanation

Self-explanatory. zzzzz is the area number.

System action

Program FABCUR1 discards the specification for the specified area, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the FILECTL control statement. Correct the error, and rerun the job.

FABC0142E	INVALID FILECTL= CONTROL CARD FILE zzzzz PREVIOUSLY SPECIFIED
------------------	--

Explanation

Self-explanatory. zzzzz is the file number.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the FILECTL control statement. Correct the error, and rerun the job.

FABC0143E	INVALID FILECTL= CONTROL CARD FILECTL=[(*) ALL] PREVIOUSLY SPECIFIED
------------------	---

Explanation

Program FABCUR1 detected a FILECTL control statement after having received the specified FILECTL specification.

System action

FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's*

Guide for details on the syntax of the FILECTL control statement. Correct the error, and rerun the job.

FABC0145E	INVALID TASKCTL= CONTROL CARD ERROR IN [NO. SUB-TASKS NO.IOVF BUFFERS UOW BUFFER SIZE] SPECIFICATION
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the TASKCTL control statement. Correct the error, and rerun the job.

FABC0146W	TASKCTL= CONTROL CARD PREVIOUSLY PROCESSED PREVIOUS [NO. SUB-TASKS NO.IOVF BUFFERS UOW BUFFER SIZE] VALUE DISCARDED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Remove the duplicate specification in subsequent executions of FABCUR1.

FABC0147E	INVALID FORMAT= CONTROL CARD SYNTAX ERROR DETECTED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, set an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for syntactical details of the FORMAT control statement. Correct the error, and rerun the job.

FABC0148E	INVALID EXITRTN= CONTROL CARD SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, set an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for syntactical details of the EXIT control statement. Correct the error, and rerun the job.

FABC0149E	INVALID EXITRTN= CONTROL CARD EXITRTN CONTROL CARD PREVIOUSLY PROCESSED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the EXITRTN control statement. Correct the error, and rerun the job.

FABC0150E	INVALID LOADCTL= CONTROL CARD SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the LOADCTL control statement. Correct the error, and rerun the job.

FABC0151W	LOADCTL= CONTROL CARD PREVIOUSLY PROCESSED PREVIOUS LOADCTL SPECIFICATION FOR SEGMENT <i>segname</i> DISCARDED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Remove the duplicate control statement in subsequent executions of FABCUR1.

FABC0152W	LOADCTL SPECIFICATION FOR [ROOT SDEP] SEGMENT (<i>segname</i>) IGNORED
------------------	---

Explanation

LOADCTL cannot be specified for ROOT or SDEP segments.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Remove the specified control statement in subsequent executions of FABCUR1.

FABC0153E	INVALID LOADCTL= CONTROL CARD TOO MANY LOADCTL SPECIFICATIONS ENCOUNTERED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the LOADCTL control statement. Correct the errors, and rerun the job.

FABC0154I	FABCUR1 NOT APF AUTHORIZED PROGRAM; ACCESS=FAST IGNORED
------------------	--

Explanation

Program FABCUR1 found that the IMS HP Fast Path Utilities load module library data set specified on the JOBLIB/STEPLIB DD statement was not authorized by APF. To invoke the ACCESS=FAST function, the load module FABCUR1 must be on the APF authorized library.

System action

FABCUR1 ignores the ACCESS=FAST request and continues processing with the ACCESS=VSAM option.

User response

Make the IMS HP Fast Path Utilities load module library APF authorized for future processing.

FABC0155W	xxxxxxxxxx; NEWDBDNM= KEYWORD IGNORED
------------------	--

Explanation

Program FABCUR1 found a DB name change requirement, but the NEWACB DD statement was not present or the IMSCATACB_OUTPUT keyword is not specified. An additional message FABC0164I is printed.

System action

FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Correct the control statement stream in subsequent executions of FABCUR1.

FABC0156E	INVALID PTRERROR= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for the syntactical details of the PTRERROR control statement. Correct the error, and rerun the job.

FABC0157W	PTRERROR= CONTROL CARD PREVIOUSLY PROCESSED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Unload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for the syntactical details of the PTRERROR control statement. Correct the error, and rerun the job.

FABC0158E	INVALID KEYSEQERROR= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

Correct the error, and rerun the job.

FABC0159W	KEYSEQERROR= CONTROL CARD PREVIOUSLY PROCESSED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal error flag, and continues processing.

User response

Correct the error, and rerun the job.

FABC0160W	xxxxxxx; HIERCHNG=YES/ YESFORCE IGNORED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Correct the control statement stream in subsequent executions of FABCUR1. The HIERCHNG keyword of the DBDNAME control statement should be specified with a value of YES/YESFORCE only when:

1. A change to the segment hierarchy is desired, *and*
2. A DMB reflecting that change is provided in the file associated with the NEWACB DD statement or the pending ACB definition in the IMS directory.

FABC0161W	HIERCHNG=YES/YESFORCE SPECIFIED; NO STRUCTURE CHANGES DETECTED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Correct the control statement stream in subsequent executions of FABCUR1. The HIERCHNG keyword of the DBDNAME control statement should be specified with a value of YES/YESFORCE only when:

1. A change to the segment hierarchy is desired, *and*
2. A DMB reflecting that change is provided in the file associated with the NEWACB DD statement.

FABC0162W	NEWDBDNM= SPECIFIED; NO DB NAME CHANGES DETECTED
------------------	---

Explanation

Program FABCUR1 found a DB name change requirement, but the new DB name was equal to the old one.

System action

FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

Correct the control statement stream in subsequent executions of FABCUR1. The NEWDBDNM keyword of the DBDNAME control statement should be specified only to change the name of the DB and to reload.

FAB00163W *xxxxxxxxx*; IGNORED DUE TO
"REORG" KEYWORD

Explanation

Program FABCUR1 detected the presence of a NEWACB DD statement or an IMSCATACB_OUTPUT keyword. When REORG is specified for the DBDNAME control statement, NEWACB DD statement and IMSCATACB_OUTPUT keyword are ignored.

System action

FABCUR1 continues processing in "reorg" mode (see the topic "Functions of DEDB Unload" > "Modes" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*).

User response

Review your JCL and control statements to verify that "reorg" mode is what you want. If you really have a new DBD, you want "change" mode. In this case, remove *REORG* from your control statement, and rerun the job.

FAB00164I *xxxxxxxxx*; "REORG" MODE
ASSUMED

Explanation

Program FABCUR1 issued an SVC 24 (DEVTYPE) specifying the ddname NEWACB. The return code specified that such a DD statement was not present in the JCL stream, hence REORG mode processing was assumed. If IMS-managed ACBs are used and the IMSCATACB_OUTPUT keyword is not specified, the program runs the job in REORG mode.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

FAB00165I **STRUCTURE CHANGE DETECTED
FOR SEGMENT *segname***
- "OLD" SEG-CD: *zz9*
- "NEW" SEG-CD: *zz9*

Explanation

Program FABCUR1 determined that the specified segment was not defined in the same place in the hierarchical structure in the DMB from the NEWACB ACB library as it was in the DMB from the OLDACB file.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

FAB00166I **DBD SPECIFICATION CHANGE(S)
DETECTED**
- NO. AREAS IN "OLD" DMB: *zz9*
- NO. AREAS IN "NEW" DMB: *zz9*

Explanation

Program FABCUR1 determined that the number of areas defined in the DMB from the NEWACB ACB library differed from the number defined in the DMB from the OLDACB file.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

FAB00168I **DATABASE NAME CHANGED**

Explanation

Program FABCUR1 found DB name change requirement. FABCUR1 creates the unload data sets with the DMB from the NEWACB library.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

FAB00169I **AUTHORIZED STRUCTURE
CHANGE(S) DETECTED**

Explanation

HIERCHNG=YES/YESFORCE was specified on the DBDNAME control statement and program FABCUR1 detected one or more structure changes.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

FAB00170W	I/O ERROR FOR OUTPUT DATA SET DDNAME <i>ddname1</i>, UNLOAD PROCESS CONTINUES WITH DATA SET DDNAME <i>ddname2</i>
------------------	--

Explanation

Program FABCUR1 issued a PUT for the *ddname1* specified. The PUT operation failed.

System action

FABCUR1 continues processing with the *ddname2* specified.

User response

None.

FAB00171W	HIERCHNG=YESFORCE SPECIFIED; AREA(S) NOT SPECIFIED MUST BE UNLOADED BY ANOTHER JOB(S).
------------------	---

Explanation

HIERCHNG=YESFORCE and RMODTYPE=S are specified but AREACTL=ALL is not specified so that only AREAs specified on the AREACTL control statement will be unloaded. AREAs of the DEDB which are not specified must be unloaded subsequently by other jobs.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing to unload AREAs specified on the AREACTL control statement.

User response

After finishing the job normally, another unload job for remaining AREAs of the DEDB which are not unloaded should be run subsequently. Otherwise the integrity of the DEDB between the AREAs will be lost.

FAB00172W	NEWDBDNM=<i>acbname</i> SPECIFIED; AREA(S) NOT SPECIFIED SHOULD BE UNLOADED BY ANOTHER JOB(S)
------------------	--

Explanation

NEWDBDNM=*acbname* and RMODTYPE=S are specified but AREACTL=ALL is not specified so that only areas specified on the AREACTL control statement are unloaded. Other jobs should unload the areas of the DEDB that are not specified.

System action

Program FABCUR1 sets on end-of-job return code of 4, and continues processing to unload areas specified on the AREACTL control statement.

User response

When you finish the job normally, run another unload job for remaining Areas of the DEDB.

FAB00173W	LRECL FOR DDNAME <i>ddname</i> IS OVERRIDDEN BY SYSTEM DETERMINED VALUE - SPECIFIED BLKSIZE TOO SMALL OR TOO LARGE - LRECL SPECIFIED: <i>xxxxxx</i> - LRECL OVERRIDDEN: <i>xxxxxx</i>
------------------	--

Explanation

This message is issued when LRECL is specified in JCL but its value is either smaller than the minimum or larger than the maximum tolerance level.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

None.

FAB00174E	- INCORRECT SEGMCTL= CONTROL CARD - SYNTAX ERROR DETECTED - "ALL" AND "SEGMENT NAME" PARAMETERS ARE MUTUALLY EXCLUSIVE - SEGMENT NAMES IN EXCESS OF 127 ARE SPECIFIED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0174E	- INCORRECT SEGMCTL= CONTROL CARD - SDEP SEGMENT <i>segname</i> IS SPECIFIED AGAINST SDEP=NO PHYSICAL ON DBDNAME CONTROL STATEMENT - SEGMENT: <i>segname</i> IS NOT DEFINED IN DMB xxxxxxxx FROM nnnnnnnn
------------------	---

Explanation

Program FABCUR1 found an incorrect *segname* in the SEGMCTL statement.

System action

FABCUR1 ends with an abend code 3728.

User response

Correct the *segname* in the SEGMCTL statement, and rerun the job.

FABC0175W	SEGMCTL= CONTROL CARD PREVIOUSLY PROCESSED FOR SEGMENT <i>segname</i>
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing. The specified segment is extracted.

User response

Remove the duplicate *segname* in the subsequent executions of FABCUR1.

FABC0176W	SEGMENT <i>segname</i> IS IGNORED BECAUSE OF SDEP=NO PHYSICAL ON DBDNAME CONTROL CARD
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 sets an end-of-job return code of 4, and continues processing. The specified segment is not extracted.

User response

Correct the *segname* or the SDEP= option on the DBDNAME control statement in the subsequent executions of FABCUR1.

FABC0179I	USER EXIT FAB11IE0 IS CALLED
------------------	-------------------------------------

Explanation

User exit routine FAB11IE0 is called.

System action

Program FABCUR1 continues processing.

User response

None. This message is informational.

FABC0180E	- INVALID LOADPLACE= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal flag, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0181E	- LOADPLACE= CONTROL CARD PREVIOUSLY PROCESSED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 discards the control statement, sets an internal flag, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0182E **INCORRECT LRECL= CONTROL CARD**
- SYNTAX ERROR DETECTED

Explanation

An incorrect LRECL was detected.

System action

Program FABCUR1 discards the control statement, sets an internal flag, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0183W **LRECL= CONTROL CARD PREVIOUSLY SPECIFIED**

Explanation

The LRECL= control card has been already specified.

System action

Program FABCUR1 discards the control statement, sets an internal flag, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0185I **RMODTYPE=S BUT RANDOMIZED TO ANOTHER AREA**

Explanation

RMODTYPE=S is specified, but the randomizer randomized a record to another area that is different from the original one.

System action

Program FABCUR1 continues processing.

User response

None. This message is informational.

FABC0186E **keyword1=value1 IS NOT ALLOWED**
- **keyword1=value1 AND keyword2=value2 ARE EXCLUSIVE**

Explanation

Keywords *keyword1* and *keyword2* cannot be specified together.

System action

FABCUR1 ends with an abend code of 3728.

User response

Correct or remove the indicated control statement.

FABC0187I **INFORMATION OF THE DB DEFINITION WAS OBTAINED FROM resource**

Explanation

This message indicates the resource (ACB library or IMS directory) where FABCUR1 obtained DMB definitions from.

System action

FABCUR1 continues processing.

User response

None. This message is informational.

FABC0188E **keyword= CONTROL CARD PREVIOUSLY PROCESSED**

Explanation

The indicated keyword cannot be specified more than once in a control statement.

System action

FABCUR1 ends with an abend code of 3728.

User response

Remove the duplicate specification and rerun the job.

FABC0189E **INVALID keyword= CONTROL CARD**

Explanation

Program FABCUR1 encountered an invalid specification while parsing the user-supplied control statement.

System action

FABCUR1 ends with an abend code of 3728.

User response

Correct the error and rerun the job.

**FABC0300I PROCESSING STARTED FOR AREA
 zzzzz (AREANAME areaname)**

Explanation

This message is generated as reload processing starts for each area.

System action

Program FABCUR3 continues processing.

User response

None. This message is informational.

**FABC0301I PROCESSING COMPLETE FOR
 AREA xxxxxx (AREANAME
 areaname) [AREA IS EMPTY]**

Explanation

This message is generated as reload processing successfully completes for the area. When this message is generated, the area has been successfully reloaded, and the data set has been closed. If the area continues no segment, the message text that specifies this is issued.

System action

Program FABCUR3 continues processing.

User response

None. This message is informational.

**FABC0302I AREA INITIALIZATION
 STATISTICS - DBDNAME: dbdname
 AREANAME: yyyy
 - ROOT PORTION:
 CI'S/UOW ADDRESSABLE BY
 RMOD: zzzzz9
 CI'S/UOW USED AS OVERFLOW:
 zzzzz9
 TOTAL CI'S/UNIT OF WORK: zzzzz9
 TOTAL UOW'S: zzzzzzzzz9
 - INDEPENDENT OVERFLOW
 PORTION:
 TOTAL CI'S: zzzzzzzzz9
 - SEQUENTIAL DEPENDENT
 PORTION:
 TOTAL CI'S: zzzzzzzzz9
 - SEQUENTIAL DEPENDENT
 PORTION: (ADSn: adsxxx)
 - TOTAL CI'S: zzzzzzzzz9**

Explanation

Program FABCUR3 does the area initialization function normally performed by the DEDB Initialization utility (DBFUMINO). This message provides the area statistics in the same format as the statistics provided by DBFUMINO. For DDNAME control statements on the areaxxx DD, the message, 'SEQUENTIAL DEPENDENT PORTION: (ADSn: adsxxx)' for each area data set is issued.

System action

FABCUR3 continues processing.

User response

None. This message is informational.

**FABC0303I VSAM "BUFND" VALUES:
 - AREA (SEQ): zz9 (OVERRIDDEN
 FROM zz9)
 - ADSn (SEQ): zz9 (OVERRIDDEN
 FROM zz9)
 - WORK (SEQ): zz9 (OVERRIDDEN
 FROM zz9)
 - WORK (DIR): zz9
 - DATA SPACE USED FOR WORK**

Explanation

This message provides statistics on the BUFND values used for the area and work data sets. Program FABCUR3 calculates default values for the number of sequential buffers. The number of direct buffers is provided by the IOVFBUF= keyword on the control statement (default = 4).

System action

FABCUR3 continues processing.

User response

None. This message is informational.

**FABC0304I NUMBER OF UNUSED IOVF CI'S:
 nnnnnnnnnn**

Explanation

This message provides statistics of unused IOVF CIs. The total number of unused IOVF CIs is specified by nnnnnnnnnn.

System action

Program FABCUR3 continues processing.

User response

None. This message is informational.

FABC0305I	RELOAD SUCCESSFULLY COMPLETED FOR ADSn: adsxxx
------------------	---

Explanation

This message is issued as reload processing successfully completes for the area data set on the adsxxx DD card. This message is generated only when area data sets control statements are specified on the areaxxx DD card.

System action

Program FABCUR3 continues processing.

User response

None. This message is informational.

FABC0305W	RELOAD BYPASSED FOR ADSn: adsxxx
------------------	---

Explanation

This message is issued when one of the area data sets adsxxx in areaxxx was not reloaded successfully. The processing for the specified area data set has been bypassed, but other ADS for the same area areaxxx is continuing.

System action

Program FABCUR3 continues processing.

User response

Copy data to the area data set on the specified adsxxx DD from the area data set that was reloaded successfully.

FABC0306I	AREA zzzzz (AREANAME: areaname) IS NOT REGISTERED IN DBRC
------------------	--

Explanation

Program FABCUR3 found that the specified area was not registered in DBRC.

System action

FABCUR3 continues processing.

User response

None. This message is informational.

FABC0307W	DATA FOR AREA zzzz9 BYPASSED
------------------	-------------------------------------

Explanation

This message is only generated when the STARTAREA keyword is present (that is, in restart situations). The message is issued for each area with data that is being bypassed. zzzz9 is the area number.

System action

Program FABCUR3 sets an end-of-job return code of 4, and continues processing.

User response

None.

FABC0308I	SDEP SEGMENTS ARE RELOCATED WITH SDEPRELOCATE=YES OPTION
------------------	---

Explanation

This message is generated when SDEP Relocation starts for each area.

System action

Program FABCUR3 continues processing.

User response

None. This message is informational.

FABC0309W	IOVF INTERVAL AT RBA: xxxxxxxx IS FULL
------------------	---

Explanation

Program FABCUR3 attempted to allocate an IOVF CI from an overflow unit, but found that all CIs within the overflow unit were in use. xxxxxxxx is the RBA of the associated directory entry.

System action

FABCUR3 issues a warning message, sets an end-of-job return code of 4, and sequentially searches the overflow directory entries to find an overflow unit with an available IOVF CI.

User response

This situation can seriously impact the online performance of the database. The database probably requires expansion and performance tuning.

FABC0310I	FABCUR3 ENDED NORMALLY
------------------	-------------------------------

Explanation

This message is generated on completion of processing by program FABCUR3. See message number FABC0310W. Also, review the other generated messages.

System action

FABCUR3 ends with a return code of 0 or 4.

User response

None. This message is informational.

FABC0310W	FABCUR3 ENDED WITH WARNINGS
------------------	------------------------------------

Explanation

This message is generated on completion of processing by program FABCUR3. The "W" level message denotes that trivial errors were encountered. Review the other generated messages, especially message number FABC0310I.

System action

FABCUR3 ends with a return code of 0 or 4.

User response

None.

FABC0310E	FABCUR3 ENDED WITH ERRORS - RELOADED FOR ALL AREAS SUCCESSFULLY BUT SOME ADS(S) BYPASSED
------------------	---

Explanation

This message is issued when the reload processing is completed successfully, but at least one process for area data set is bypassed. The detail of the cause is shown in the other error message for the area data set.

System action

Program FABCUR3 ends with return code of 8.

User response

Copy data to the bypassed area data set from the area data set that was reloaded successfully.

FABC0311I	DBRC=Y IS SPECIFIED
------------------	----------------------------

Explanation

DBRC=Y is specified in the EXEC parameter of the FABCUR3 JCL. Program FABCUR3 will establish a DBRC interface and obtain area information from DBRC.

System action

FABCUR3 continues processing.

User response

None. This message is informational.

FABC0312I	DBRC=N IS SPECIFIED
------------------	----------------------------

Explanation

DBRC=N is specified in the EXEC parameter of the FABCUR3 JCL. Program FABCUR3 does not establish DBRC interface.

System action

FABCUR3 continues processing.

User response

None. This message is informational.

FABC0313I	CARD xx: zzzz...zzzz
------------------	-----------------------------

Explanation

This message is shows the control statement currently being processed.

System action

Program FABCUR3 continues processing.

User response

None. This message is informational.

FABC0315I	zzz,zzz,zz9 SEGMENT RECORDS OF INSERT LIMIT COUNT IN UNLOADED FILE
------------------	---

Explanation

The Insert Limit Count (ILC) records were detected. These records are ignored in the key field sequence check.

System action

Program FABCUR3 continues processing.

User response

None. This message is informational.

FABC0316W	SUMMARY OF KEY SEQUENCE ERRORS - NUMBER OF RELATED DB RECORDS: <i>zzz,zzz,zz9</i> - NUMBER OF THE SEGMENTS DETECTED AS KEY SEQUENCE ERROR: <i>zzz,zzz,zz9</i> - ERROR SEGMENTS SUM TOTAL INCLUDING CHILD SEGMENTS: <i>zzz,zzz,zz9</i>
------------------	---

Explanation

This message means the number of the error segments of key field sequence check. This message is issued only when KEYSEQERROR=BYPASS is specified in the SYSIN DD control statement, and when FABCUR3 detected the error segments of key field sequence check.

System action

Program FABCUR3 sets an end-of-job return code of 4, and continues processing.

User response

Verify the unloaded file in the DURDATA DD statement. Correct the problem, and if necessary, rerun the unload job.

FABC0317I	EXIT ROUTINE <i>exitname</i> "END" CALL FINISHED - first 80 bytes characters of the message that user exit routine returned - subsequent 48 bytes characters of the message that user exit routine returned
------------------	--

Explanation

Program FABCUR3 called the user exit routine *exitname* with "END" call and the exit routine returned the message specified.

System action

FABCUR3 continues processing.

User response

None. This message is informational.

FABC0318I	SDEPRELOCATE=YES OPTION IS IGNORED
------------------	---

Explanation

Program FABCUR3 has been ignored for one of the following reasons:

- The unloaded file that was used as the input of Reload processing was not unloaded with the SDEP=PHYSICAL option.
- The SDEP Logical Begin and Logical End pointers in the second CI of the original area were not physically inverted.
- The target area has insufficient SDEP space for SDEP Relocation.

System action

FABCUR3 will not relocate SDEP pointers, and continues processing.

User response

Check the condition for SDEP relocation; if necessary, correct the condition; rerun the job.

FABC0320E	UNKNOWN KEYWORD (NEAR COLUMN <i>xx</i>)
------------------	--

Explanation

Program FABCUR3 was searching for the start of a "keyword=value" specification on the control statement. At column *xx* of the statement, an unknown keyword was encountered.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0321E	"STARTAREA=" VALUE INVALID
------------------	-----------------------------------

Explanation

Program FABCUR3 encountered a STARTAREA keyword whose associated parameter value is missing or not numeric.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0322E "IOVFBUF=" VALUE INVALID

Explanation

Program FABCUR3 encountered an IOVFBUF= keyword whose associated parameter value is missing or not numeric.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

**FABC0323W "IOVFBUF=" LESS THAN MIN.
REQUIRED - DEFAULT ASSUMED**

Explanation

Program FABCUR3 encountered an IOVFBUF= keyword whose associated parameter value was less than the minimum acceptable value (default = 4).

System action

FABCUR3 assumes the default minimum, sets an end-of-job return code of 4, and continues processing.

User response

None.

FABC0324E "TBLENTY=" VALUE INVALID

Explanation

Program FABCUR3 encountered a TBLENTY= keyword whose associated parameter value is missing or not numeric.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0325E "EXITRTN" VALUE INVALID

Explanation

Self explanatory.

System action

Program FABCUR3 discards the control statement, sets an internal error flag, and continues processing.

User response

See the topic "Input for DEDB Reload" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details on the syntax of the EXITRTN control statement. Correct the error, and rerun the job.

**FABC0326E "EXITRTN=" PREVIOUSLY
SPECIFIED**

Explanation

Self-explanatory.

System action

Program FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

**FABC0327I - "SDEPRELOCATE=YES" IS
IGNORED DUE TO AREC=N**

Explanation

Program FABCUR3 found SDEPRELOCATE=YES in the SYSIN DD statement when AREC=N was specified on the EXEC parameter.

System action

FABCUR3 ignores this keyword and will not relocate SDEP pointers. The reload process continues.

User response

Check the condition for SDEP relocation. If necessary, correct the condition and rerun the job.

**FABC0328I SDEP SEGMENTS ARE RELOCATED
DUE TO SDEP=PHYSICAL
SPECIFIED WITH xxxxxxxx**

Explanation

Locations of each SDEP segments are changed from the original RBA because the segments were unloaded

with SDEP=PHYSICAL accompanied by a DEDB change with NEWACB or the IMSCATACB_OUTPUT keyword.

System action

Program FABCUR3 will relocate the RBA of each SDEP segments, associating RBA of PCF pointer in their root segments, and PTF pointer of their twin segments.

User response

The RBA value to identify SDEP marker is no longer used because absolute value of RBA of each SDEP segments at unload were changed in reloaded area.

FABC0330E	areaxxx CONTROL STATEMENT DATASET IS EMPTY
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR3 ends with an abend code of 3761.

User response

Verify the correctness of the control statement on the areaxxx DD. Correct the error, and rerun the job.

FABC0331E	UNKNOWN KEYWORD ENCOUNTERED IN areaxxx DATASET
------------------	---

Explanation

Program FABCUR3 found that there were invalid string on the areaxxx DD control statement.

System action

FABCUR3 ends with an abend code of 3761.

User response

Verify the correctness of the control statement on the areaxxx DD. Correct the error, and rerun the job.

FABC0332E	NO VALID DDNAME CONTROL STATEMENT SPECIFIED IN areaxxx DATASET
------------------	---

Explanation

Program FABCUR3 could not find valid DDNAME control statement on the areaxxx DD.

System action

FABCUR3 ends with an abend code of 3761.

User response

Verify the correctness of the control statement on the areaxxx DD. Correct the error, and rerun the job.

FABC0333E	MORE THAN 7 DDNAME CONTROL STATEMENTS SPECIFIED IN areaxxx DATASET
------------------	---

Explanation

Program FABCUR3 found that there were more than 7 DDNAME control statements on the areaxxx DD.

System action

FABCUR3 ends with an abend code of 3761.

User response

Verify the correctness of the control statement on the areaxxx DD. Correct the error, and rerun the job.

FABC0334E	DUPLICATE DDNAME adsxxx SPECIFIED IN areaxxx DATASET
------------------	---

Explanation

Program FABCUR3 found that duplicate DDNAME control statements were specified on one areaxxx DD.

System action

FABCUR3 ends with an abend code of 3761.

User response

Verify the correctness of the control statement on the areaxxx DD. Correct the error, and rerun the job.

FABC0335E	NO DDNAME SPECIFIED IN areaxxx DATASET
------------------	---

Explanation

Program FABCUR3 found that neither VSAM data set nor DDNAME control statement was specified on the areaxxx DD.

System action

FABCUR3 ends with an abend code of 3761.

User response

Verify the correctness of the control statement on the areaxxx DD. Correct the error, and rerun the job.

FABC0336E "IMGCPY=" VALUE INVALID

Explanation

Program FABCUR3 encountered an IMGCPY= keyword whose associated parameter value is missing or incorrect.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0342E "ICCOMPRESS=" VALUE INVALID

Explanation

Program FABCUR3 encountered an ICCOMPRESS= keyword whose associated parameter value is missing or incorrect.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0343E "ICHASH=" VALUE INVALID

Explanation

Program FABCUR3 encountered a ICHASH= keyword whose associated parameter value is missing or incorrect.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

**FABC0344E "IMGCPY=" PREVIOUSLY
SPECIFIED**

Explanation

Self-explanatory.

System action

Program FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

**FABC0345E "ICCOMPRESS=" PREVIOUSLY
SPECIFIED**

Explanation

Self-explanatory.

System action

Program FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

**FABC0346E "ICHASH=" PREVIOUSLY
SPECIFIED**

Explanation

Self-explanatory.

System action

Program FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

**FABC0347E ICHASH=YES AND/OR
ICCOMPRESS=YES SPECIFIED
BUT IMGCPY=YES|DUAL NOT
SPECIFIED**

Explanation

Program FABCUR3 found that there was no IMGCPY=YES|DUAL keyword parameter specified even though ICHASH=(YES), ICCOMPRESS=(YES), or both were specified. To invoke the ICHASH=YES option and/or the ICCOMPRESS=YES option, IMGCPY=YES|DUAL must also be specified.

System action

FABCUR3 ends with an abend code of 3761.

User response

Correct the control statement, and rerun the job.

FABC0348W	"OPEN" FAIL FOR DDNAME [HDIxxxxy/]Xlxxxxxy - DD STATEMENT NOT FOUND OR DUMMY SPECIFIED
------------------	---

Explanation

Program FABCUR3 found that a DD statement was not specified or DD DUMMY was specified for the ddname specified to create an image copy.

System action

FABCUR3 ignores image copy processing for the associating area with an area number specified by xxx or xxxxx, and continues unload operation for the succeeding areas.

User response

If image copy is required for the area, run the IMS Image Copy utility or the IBM IMS Image Copy Extensions for z/OS utility after the job is finished.

FABC0350I	IMAGE COPY PROCESSING STARTED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR3 starts an image copy subtask.

User response

None. This message is informational.

FABC0351I	- IMAGE COPY PROCESSING ENDED NORMALLY
------------------	---

Explanation

This message is generated when all requested image copy processing are completed without errors.

System action

If there were no other FABC03xxW/E messages, program FABCUR3 will end with the return code of 0.

User response

None. This message is informational.

FABC0351W	- IMAGE COPY PROCESSING ENDED WITH WARNINGS RC=xx
------------------	--

Explanation

This message is generated when one or more requested image copy processing has been completed with errors. xx shows the highest return codes that the IBM IMS Image Copy Extensions for z/OS utility returned in message FABC0353W.

System action

If there were no other FABC03xxE messages, program FABCUR3 will end with a return code of 4.

User response

Follow the programmer action for message FABC0353W.

FABC0351E	- IMAGE COPY PROCESSING ENDED ABNORMALLY
------------------	---

Explanation

This message is generated when the program FABCUR3 main task detected critical image copy process errors, including image copy subtask abends, when finishing the reload operation for all areas.

System action

If there were no other FABC03xxE messages, FABCUR3 will end with a return code of 4.

User response

None.

FABC0352I	- IMAGE COPY COMPLETED FOR AREA zzzzz (AREANAME areaname) TIME STAMP - xx...xx COPY 1 DATASET NAME - dd...dd COPY 2 DATASET NAME - dd...dd
------------------	---

Explanation

Image copy processing for the area specified completed normally. The time stamp of Image copy and the image copy data set(s) are shown.

System action

Program FABCUR3 continues.

User response

None. This message is informational.

**FABC0353W - IMAGE COPY COMPLETED WITH
 ERRORS FOR AREA zzzzzz (AREA
 NAME:areaname) RC=xx**

Explanation

Image copy processing for the area specified completed, and the IBM IMS Image Copy Extensions for z/OS utility returned error return code xx.

System action

Program FABCUR3 continues.

User response

Follow the programmer action of any messages issued by the IBM IMS Image Copy Extensions for z/OS utility.

**FABC0354E "KEYSEQERROR=" PREVIOUSLY
 SPECIFIED**

Explanation

Self-explanatory.

System action

Program FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

**FABC0355E "KEYSEQERROR=" VALUE
 INCORRECT**

Explanation

Program FABCUR3 encountered a KEYSEQERROR= keyword whose associated parameter value is missing or incorrect.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0356E "SDEPRELOCATE= OR SDEPRE="
PREVIOUSLY SPECIFIED

Explanation

Self-explanatory.

System action

Program FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0357E "SDEPRELOCATE= OR SDEPRE="
VALUE INCORRECT

Explanation

Program FABCUR3 encountered an SDEPRELOCATE= keyword whose associated parameter value is missing or incorrect.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing

User response

Correct the control statement, and rerun the job.

FABC0360I USER EXIT FABC3IE0 IS CALLED

Explanation

User exit routine FABC3IE0 is called.

System action

Program FABCUR3 continues processing.

User response

None. This message is informational.

FABC0361I INFORMATION OF THE DB
DEFINITION WAS OBTAINED
FROM resource

Explanation

This message is to inform which resource (DURDBDFN DD, ACBLIB DD, or the IMS directory) is used to obtain the DEDB definition information.

System action

Program FABCUR3 continues processing.

User response

None. This message is informational.

FABC0362E	"RAPERROR=" PREVIOUSLY SPECIFIED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0363E	"RAPERROR=" VALUE INCORRECT
------------------	------------------------------------

Explanation

Program FABCUR3 encountered a RAPERROR= keyword whose associated parameter value is missing or incorrect.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0364E	"DBDNAME=" PREVIOUSLY SPECIFIED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0365E	"DBDNAME=" VALUE INCORRECT
------------------	-----------------------------------

Explanation

Program FABCUR3 encountered a DBDNAME= keyword whose associated parameter value is missing or incorrect.

System action

FABCUR3 sets an end-of-job return code of 8, and continues processing.

User response

Correct the control statement, and rerun the job.

FABC0368E	"keyword=" PREVIOUSLY SPECIFIED
------------------	--

Explanation

The indicated keyword cannot be specified more than once in a control statement.

System action

FABCUR3 ends with an abend code of 3728.

User response

Remove the duplicate specification and rerun the job.

FABC0369E	"keyword=" VALUE INCORRECT
------------------	-----------------------------------

Explanation

Program FABCUR3 encountered an invalid specification while parsing the user-supplied control statement.

System action

FABCUR3 ends with an abend code of 3761.

User response

Correct the error and rerun the job.

FABC0370I	INSUFF. STORAGE FOR: aaaa - INCREASE REGION SIZE
------------------	---

Explanation

Program FABCUR3 issued a GETMAIN macro to allocate storage for the purpose of aaaa. The attempt was unsuccessful.

System action

FABCUR3 calculates HIGH ALLOCATE RBA of the multi-volume ADS without using the Catalog Search Interface (CSI) and continues processing.

User response

None. This message is informational.

FABC0371I	FAILURE READING DATA SET INFORMATION FROM CATALOG - DSN: <i>data_set</i> - REASON CODE: <i>rsn</i> RETURN CODE: <i>rc</i> - NOT FOUND DATA PORTION - INCONSISTENT ENTRY NUMBER. VOLSER: <i>nn xxxxxxxx: nn</i> - THE NUMBER OF VOLUME IS BEYOND THE LIMIT
------------------	--

Explanation

An inconsistency is found in the catalog information.

System action

FABCUR3 calculates HIGH ALLOCATE RBA of the multi-volume ADS without using the Catalog Search Interface (CSI) and continues processing.

User response

None. This message is informational.

FABC0500I	FABCUR5 ENDED NORMALLY
------------------	-------------------------------

Explanation

This message is generated when all requested processing has been completed without errors.

System action

Program FABCUR5 ends with a return code of 0.

User response

None. This message is informational.

FABC0500W	FABCUR5 ENDED WITH WARNINGS
------------------	--

Explanation

This message is generated when trivial error conditions were encountered by program FABCUR5.

System action

FABCUR5 ends with a return code of 4.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR5. Correct the problem and rerun the job, or continue with the processing, as desired.

FABC0500E	FABCUR5 ENDED WITH ERRORS
------------------	----------------------------------

Explanation

This message is generated when nontrivial error conditions were encountered by program FABCUR5.

System action

FABCUR5 ends with a return code of 8.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR5. Correct the problem and rerun the job.

FABC0501I	DATABASE DEFINITION RECORD FOR DBD: <i>dbdname</i> IS BUILT
------------------	--

Explanation

This message is generated when program FABCUR5 built the database definition record for the DBD specified.

System action

FABCUR5 continues processing.

User response

None. This message is informational.

FABC0502I	DBD DEFINITION INFORMATION FOR DBD: <i>dbdname</i> IS REPORTED
------------------	---

Explanation

This message is generated when program FABCUR5 reports the DBD definition information for the DBD specified.

System action

FABCUR5 continues processing.

User response

None. This message is informational.

FABC0520I CARD xx: zzzz...zzzz

Explanation

This message is generated to show the control statement currently being processed.

System action

Program FABCUR5 continues processing.

User response

None. This message is informational.

**FABC0521W ERROR DETECTED NEAR COLUMN
 xx**

Explanation

Program FABCUR5 detected an error in the control statement currently being processed. (See the immediately preceding FABC0520I message.)

System action

FABCUR5 continues processing, and issues one or more other FABC05xx messages.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR5. Correct the problem and rerun the job, or continue with reload processing, as desired.

**FABC0521E ERROR DETECTED NEAR COLUMN
 xx**

Explanation

Program FABCUR5 detected an error in the control statement currently being processed. (See the immediately preceding FABC0520I message.)

System action

FABCUR5 continues processing, and issues one or more other FABC05xx messages.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR5. Correct the problem and rerun the job, or continue with reload processing, as desired.

FABC0522W BLANK/INVALID CONTROL CARD

Explanation

Self-explanatory

System action

Program FABCUR5 discards the control statement, sets an end-of-job return code of 4, and continues processing.

User response

Remove the specified control statement in subsequent execution of FABCUR5.

FABC0523E UNKNOWN KEYWORD

Explanation

Program FABCUR5 encountered a control statement with a value starting in column one that is not one of the valid control statements.

System action

FABCUR5 ends with an abend code of 3728.

User response

Correct, or remove, the specified control statement.

**FABC0525E 1ST CONTROL CARD NOT
 DBDNAME= CARD**

Explanation

Self-explanatory.

System action

Program FABCUR5 ends with an abend code of 3728.

User response

The control statement stream must include one DBDNAME control statement, and it must be the first statement in the stream. Correct the control statement stream. Rerun the job.

**FABC0526E INVALID DBDNAME= CONTROL
 CARD
 - MULTIPLE DBDNAME= CARDS
 ENCOUNTERED**

Explanation

Self-explanatory.

System action

Program FABCUR5 ends with an abend code of 3728.

User response

The control statement stream must include one DBDNAME control statement and it must be the first statement in the stream. Correct the control statement stream, and rerun the job.

FABC0527E	INVALID DBDNAME= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR5 discards the control statement, sets an internal error flag, and continues processing.

User response

See [“Input for the Database Definition Record Create utility” on page 28](#) for details on the syntax of the DBDNAME control statement. Correct the error, and rerun the job.

FABC0528E	INVALID DBDNAME= CONTROL CARD - DBDNAME MISSING/INVALID
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR5 discards the control statement, sets an internal error flag, and continues processing.

User response

See [“Input for the Database Definition Record Create utility” on page 28](#) for details on the syntax of the DBDNAME control statement. Correct the error, and rerun the job.

FABC0530E	INVALID FUNCTION= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR5 discards the control statement, sets an internal error flag, and continues processing.

User response

See [“Input for the Database Definition Record Create utility” on page 28](#) for details on the syntax of the FUNCTION control statement. Correct the error, and rerun the job.

FABC0531E	INVALID FUNCTION= CONTROL CARD - "FUNCTION=" CONTROL CARD PREVIOUSLY SPECIFIED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR5 discards the control statement, sets an internal error flag, and continues processing.

User response

See [“Input for the Database Definition Record Create utility” on page 28](#) for details on the syntax of the FUNCTION control statement. Correct the error, and rerun the job.

FABC0540I	INPUT= CONTROL CARD IGNORED
------------------	------------------------------------

Explanation

Self-explanatory.

System action

Program FABCUR5 discards the control statement and continues processing.

User response

None. This message is informational.

FABC0541E	NO DD OR DUMMY SPECIFIED FOR BOTH ACBLIB DD AND DURDBDFN DD
------------------	--

Explanation

Program FABCUR5 found that no DD statement or DUMMY is specified for both ACBLIB DD and DURDBDFN DD data sets for FUNCTION=PRINT request.

System action

FABCUR5 ends with a return code of 8.

User response

Specify the correct data set for the ACBLIB DD, or the DURDBDFN DD statements, or both, and rerun the job.

FAB0600I FABCUR6 ENDED NORMALLY

Explanation

This message is generated when all requested processing has been completed without errors.

System action

Program FABCUR6 finished 'EOF' function with no errors.

User response

None. This message is informational.

FAB0600W FABCUR6 ENDED WITH WARNING

Explanation

This message is generated when trivial error conditions were encountered by program FABCUR6.

System action

FABCUR6 finished 'EOF' function with trivial errors.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR6. Correct the problem and rerun the job, or continue with the processing, as desired.

FAB0600E FABCUR6 ENDED WITH ERRORS

Explanation

This message is generated when nontrivial error conditions were encountered by program FABCUR6.

System action

FABCUR6 finished 'EOF' function with nontrivial errors.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR6. Correct the problem, and rerun the job.

**FAB0601I OBTAINED DB DEFINITIONS
FROM *resource***

Explanation

This message indicates the *resource* (ACB library or IMS directory) where FABCUR6 obtained database definitions from.

System action

FABCUR6 continues processing.

User response

None. This message is informational.

FAB0602W NO 'PUT' CALL PROCESSED

Explanation

This message is generated when the application program ended without a 'PUT' call request to program FABCUR6.

System action

FABCUR6 ends with FAB0600W message.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR6. Correct the problem and rerun the job.

**FAB0603W NO SEGMENTS WILL BE
RELOADED TO AREA zzzzz
(AREaname: yyyyyyyy)
- EMPTY AREA WILL NOT BE
RELOADED BECAUSE EMPTY=NO
IS SPECIFIED
- EMPTY AREA WILL NOT BE
RELOADED BECAUSE FILECTL
STATEMENT IS NOT SPECIFIED
FOR THIS AREA
- EMPTY AREA WILL BE
RELOADED**

Explanation

There is no segment record written for area zzzzz. One of the three sub-texts follows the FAB0603W message to indicate that an empty area for area zzzzz will or will not be reloaded by the succeeding reload (FABCUR3) process.

System action

Program FABCUR6 continues processing.

User response

Attempt to determine if there should have been any segment data records written to the specified output file. Verify that the DD statement ACBLIB/IMSACBA/IMSACBB correctly identifies the proper data set, and that the DBDGEN and ACBGEN for the database being processed were performed correctly. Check that the randomizer module is specified correctly. Review the FILECTL specifications, if any. If the condition described by the sub-text for the empty area is not an expected result, then check that the EMPTY= option and the FILECTL statement(s) are specified correctly. Correct the problem and rerun the job, or continue with reload processing, as desired.

FABC0604W	NO RECORDS WRITTEN TO DDNAME DURDzzzO / XDzzzzzO EXCEPT AREA INFORMATION RECORD
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR6 continues processing.

User response

Attempt to determine if there should have been any segment data records written to the specified output file. Verify that the DD statement ACBLIB/IMSACBA/IMSACBB correctly identifies the proper data set, and that the DBDGEN and ACBGEN for the database being processed were performed correctly. Check that the randomizer module is specified correctly. Review the FILECTL specifications, if any. Correct the problem and rerun the job, or continue with reload processing, as desired.

FABC0605W	EMPTY=NO IS FORCED BECAUSE AREA_INFORMATION_RECORD=N O IS SPECIFIED
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR6 continues processing.

User response

When AREA_INFORMATION_RECORD=NO is specified, EMPTY=YES is overridden by EMPTY=NO unless both FORMAT=TFMT and LRECL=SEGTFMT are

specified with EMPTY=YES. Check that the EMPTY option and the AREA_INFORMATION_RECORD option are specified correctly. Correct the problem and rerun the job, or continue with reload processing, as desired.

FABC0611I	EXIT ROUTINE <i>exitname</i> "END" CALL FINISHED - first 80 bytes characters of the message that user exit routine returned - subsequent 48 bytes characters of the message that user exit routine returned
------------------	--

Explanation

Program FABCUR6 called the user exit routine *exitname* with an "END" call and the exit routine returned the message specified.

System action

FABCUR6 continues processing.

User response

None. This message is informational.

FABC0620I	CARD xx: zzzz...zzzz
------------------	-----------------------------

Explanation

This message is generated to show the control statement currently being processed.

System action

Program FABCUR6 continues processing.

User response

None. This message is informational.

FABC0621W	ERROR DETECTED NEAR COLUMN xx
------------------	--

Explanation

Program FABCUR6 detected an error in the control statement currently being processed. (See the immediately preceding FABC0620I message.)

System action

FABCUR6 continues processing, and issues one or more other FABC06xx messages.

User response

To determine the nature and causes of the errors detected, see the other messages generated by FABCUR6. Correct the problem and rerun the job, or continue with reload processing, as desired.

FABC0621E	ERROR DETECTED NEAR COLUMN XX
------------------	--

Explanation

See message number FABC0621W.

System action

See message number FABC0621W.

User response

See message number FABC0621W.

FABC0622W	BLANK/INVALID CONTROL CARD
------------------	-----------------------------------

Explanation

Self-explanatory.

System action

Program FABCUR6 discards the control statement, sets an end-of-job return code of 4, and continues processing.

User response

Remove the specified control statement in subsequent executions of FABCUR6.

FABC0623E	UNKNOWN KEYWORD
------------------	------------------------

Explanation

Program FABCUR6 encountered a control statement with a value starting in column one that is not one of the valid control statements.

System action

FABCUR6 ends with an abend code of 3728.

User response

Correct, or remove, the specified control statement.

FABC0640E	INVALID FILECTL= CONTROL CARD - FILE NO(S) SPECIFICATION MISSING/INVALID
------------------	---

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the FILECTL control statement. Correct the control statement stream, and rerun the job.

FABC0641E	INVALID FILECTL= CONTROL CARD - AREA zzzzz PREVIOUSLY SPECIFIED
------------------	--

Explanation

Self-explanatory. zzzzz is the area number.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the FILECTL control statement. Correct the error, and rerun the job.

FABC0642E	INVALID FILECTL= CONTROL CARD - FILE zzzzz PREVIOUSLY SPECIFIED
------------------	--

Explanation

Self-explanatory. zzzzz is the file number.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the FILECTL control statement. Correct the error, and rerun the job.

FABC0643E	INVALID FILECTL= CONTROL CARD - FILECTL=[(*) ALL] PREVIOUSLY SPECIFIED
------------------	---

Explanation

Program FABCUR6 detected a FILECTL control statement after having received the specified FILECTL specification.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the FILECTL control statement. Correct the error, and rerun the job.

FABC0644E	INVALID EXITRTN= CONTROL CARD
	- SYNTAX ERROR DETECTED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details on the syntax of the EXITRTN control statement. Correct the error, and rerun the job.

FABC0645E	INVALID EXITRTN= CONTROL CARD
	- "EXITRTN=" CONTROL CARD PREVIOUSLY SPECIFIED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the EXITRTN control statement. Correct the error, and rerun the job.

FABC0646E	INVALID EXITRTN= CONTROL CARD
------------------	--------------------------------------

- EXITRTN NAME SPECIFIED TOO LONG

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the EXITRTN control statement. Correct the error, and rerun the job.

FABC0647E	INVALID IMSCOMP= CONTROL CARD
	- SYNTAX ERROR DETECTED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the IMSCOMP control statement. Correct the error, and rerun the job.

FABC0648E	INVALID IMSCOMP= CONTROL CARD
	- "IMSCOMP=" CONTROL CARD PREVIOUSLY SPECIFIED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the IMSCOMP control statement. Correct the error, and rerun the job.

FABC0649E	INVALID USERCTL= CONTROL CARD
------------------	--------------------------------------

- SYNTAX ERROR DETECTED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See [“Input for the DEDB Reload Segment Data Set Create utility” on page 42](#) for details about the syntax of the USERCTL control statement. Correct the error, and rerun the job.

FABC0654E	INVALID FORMAT= CONTROL CARD
	- SYNTAX ERROR DETECTED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See [“Input for the DEDB Reload Segment Data Set Create utility” on page 42](#) for details about the syntax of the FORMAT control statement. Correct the error, and rerun the job.

FABC0655E	INVALID FORMAT= CONTROL CARD
	- "FORMAT=" CONTROL CARD PREVIOUSLY SPECIFIED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See [“Input for the DEDB Reload Segment Data Set Create utility” on page 42](#) for details about the syntax of the FORMAT control statement. Correct the error, and rerun the job.

FABC0656E	- INVALID AREA_INFORMATION_RECORD= CONTROL CARD
------------------	--

- SYNTAX ERROR DETECTED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

For details on the syntax of the AREA_INFORMATION_RECORD control statement, see [“Input for the DEDB Reload Segment Data Set Create utility” on page 42](#). Correct the error, and rerun the job.

FABC0657E	- INVALID AREA_INFORMATION_RECORD= CONTROL CARD
	- "AREA_INFORMATION_RECORD=" CONTROL CARD PREVIOUSLY SPECIFIED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

For details on the syntax of the AREA_INFORMATION_RECORD control statement, see [“Input for the DEDB Reload Segment Data Set Create utility” on page 42](#). Correct the error, and rerun the job.

FABC0658E	- INVALID LRECL= CONTROL CARD
	- SYNTAX ERROR DETECTED

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

Correct the control statement, and rerun the job.

FABC0659E	- INVALID LRECL= CONTROL CARD
------------------	--------------------------------------

- "LRECL=" CONTROL CARD
PREVIOUSLY SPECIFIED

Explanation

The LRECL= control card has been already specified.

System action

FABCUR6 ends with an abend code of 3728.

User response

Correct the control statement, and rerun the job.

FABC0660W	- [BLKSIZE LRECL] FOR DDNAME <i>ddname</i> IS OVERRIDDEN BY SYSTEM DETERMINED VALUE - SPECIFIED [BLKSIZE LRECL] TOO SMALL OR TOO LARGE - [BLKSIZE LRECL] SPECIFIED: <i>mmmmm</i> - [BLKSIZE LRECL] OVERRIDDEN: <i>nnnnn</i>
------------------	--

Explanation

This message is issued when BLKSIZE/LRECL is specified in JCL but its value is either smaller than the minimum or larger than the maximum tolerance level.

System action

Program FABCUR6 continues processing.

User response

None.

FABC0661E	WHEN LRECL=SEGTFMT IS SPECIFIED, FORMAT=TFMT HAS TO BE SPECIFIED
------------------	---

Explanation

You have to specify FORMAR=TFMT when LRECL=SEGTFMT is specified.

System action

Program FABCUR6 sets an internal error flag, and continues processing.

User response

Correct the error, and rerun the job.

FABC0670W	I/O ERROR FOR OUTPUT DATA SET DDNAME <i>ddname1</i>, UNLOAD
------------------	--

PROCESS CONTINUES WITH DATA
SET DDNAME *ddname2*

Explanation

Program FABCUR6 issued a PUT for the *ddname1* specified. The PUT operation failed.

System action

FABCUR6 continues processing with the specified *ddname2*.

User response

None.

FABC0672E	INVALID OUTDD= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See [“Input for the DEDB Reload Segment Data Set Create utility”](#) on page 42 for details about the syntax of the OUTDD control statement. Correct the error, and rerun the job.

FABC0673E	INVALID OUTDD= CONTROL CARD - "OUTDD=" CONTROL CARD PREVIOUSLY SPECIFIED
------------------	---

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See [“Input for the DEDB Reload Segment Data Set Create utility”](#) on page 42 for details about the syntax of the OUTDD control statement. Correct the error, and rerun the job.

FABC0674E	INVALID IMSCATHLQ= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the USERCTL control statement. Correct the error and rerun the job.

FAB0675E	INVALID IMSCATHLQ= CONTROL CARD - "IMSCATHLQ=" CONTROL CARD PREVIOUSLY SPECIFIED
-----------------	---

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the FILECTL control statement. Correct the error and rerun the job.

FAB0676E	INVALID IMSCATHLQ= CONTROL CARD - IMSCATHLQ SPECIFIED TOO LONG
-----------------	---

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the EXITRTN control statement. Correct the error and rerun the job.

FAB0677E	INVALID IMSCATACB_INPUT= CONTROL CARD - SYNTAX ERROR DETECTED
-----------------	--

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the USERCTL control statement. Correct the error and rerun the job.

FAB0678E	INVALID IMSCATHLQ_INPUT= CONTROL CARD - "IMSCATHLQ_INPUT=" CONTROL CARD PREVIOUSLY SPECIFIED
-----------------	---

Explanation

Self-explanatory.

System action

FABCUR6 ends with an abend code of 3728.

User response

See “Input for the DEDB Reload Segment Data Set Create utility” on page 42 for details about the syntax of the FILECTL control statement. Correct the error and rerun the job.

FAB0700I	FABCUR7 ENDED NORMALLY
-----------------	-------------------------------

Explanation

This message is generated when all requested processing has been completed without errors.

System action

Program FABCUR7 finished 'EOF' function with no errors.

User response

None. This message is informational.

FAB0700W	FABCUR7 ENDED WITH WARNING
-----------------	-----------------------------------

Explanation

This message is generated when trivial error conditions were encountered by program FABCUR7.

System action

FABCUR7 finished 'EOF' function with trivial errors.

User response

See the other messages generated by FABCUR7 to determine the nature and causes of the errors detected. Correct the problem and rerun the job, or continue with the processing, as desired.

FABC0700E	FABCUR7 ENDED WITH ERRORS
------------------	----------------------------------

Explanation

This message is generated when nontrivial error conditions were encountered by program FABCUR7.

System action

FABCUR7 finished 'EOF' function with nontrivial errors.

User response

See the other messages generated by FABCUR7 to determine the nature and causes of the errors detected. Correct the problem, and rerun the job.

FABC0701E	INVALID EXITRTN= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61 for details about the syntax of the EXITRTN control statement. Correct the error, and rerun the job.

FABC0702E	INVALID EXITRTN= CONTROL CARD - "EXITRTN=" CONTROL CARD PREVIOUSLY SPECIFIED
------------------	---

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61 for details about the syntax of the EXITRTN control statement. Correct the error, and rerun the job.

FABC0703E	INVALID EXITRTN= CONTROL CARD - EXITRTN NAME SPECIFIED TOO LONG
------------------	--

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61 for details about the syntax of the EXITRTN control statement. Correct the error, and rerun the job.

FABC0704E	INVALID IMSCOMP= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61 for details about the syntax of the IMSCOMP control statement. Correct the error, and rerun the job.

FABC0705E	INVALID IMSCOMP= CONTROL CARD - "IMSCOMP=" CONTROL CARD PREVIOUSLY SPECIFIED
------------------	---

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61 for details about the syntax of the IMSCOMP control statement. Correct the error, and rerun the job.

FABC0706I **CARD xx: zzzz...zzzz**

Explanation

This message is generated to show the control statement currently being processed.

System action

Program FABCUR7 continues processing.

User response

None. This message is informational.

FABC0707W **ERROR DETECTED NEAR COLUMN
xx**

Explanation

Program FABCUR7 detected an error in the control statement currently being processed. (See the immediately preceding FABC0706I message.)

System action

FABCUR7 continues processing, and issues one or more other FABC07xx messages.

User response

See the other messages generated by FABCUR7 to determine the nature and causes of the errors detected. Correct the problem and rerun the job, or continue with reload processing, as desired.

FABC0708W **BLANK/INVALID CONTROL CARD**

Explanation

Self-explanatory.

System action

Program FABCUR7 discards the control statement, sets an internal error flag, and continues processing.

User response

Remove the specified control statement in subsequent executions of FABCUR7.

FABC0709E **UNKNOWN KEYWORD**

Explanation

Program FABCUR7 encountered a control statement with a value starting in column one that is not one of the valid control statement types.

System action

FABCUR7 ends with an abend code of 3728.

User response

Correct, or remove, the specified control statement, and rerun the job.

FABC0710W **NO RECORD PROVIDED FROM
UR7DATA/UR7DATA1/UR7DATA2**

Explanation

Program FABCUR7 found that there was no record read from the specified unloaded segment data set.

System action

FABCUR7 will issue FABC0700W message.

User response

Verify that the correct unloaded segment data set was specified.

FABC0711I **EXIT ROUTINE *exitname* "END"
CALL FINISHED**
- first 80 bytes characters of the
message that user exit routine
returned
- subsequent 48 bytes characters
of the message that user exit
routine returned

Explanation

Program FABCUR7 called the user exit routine *exitname* with "END" call and the exit routine returned the message specified.

System action

FABCUR7 continues.

User response

None. This message is informational.

FABC0714E **- INVALID
AREA_INFORMATION_RECORD=
CONTROL CARD
- SYNTAX ERROR DETECTED**

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

For details on the syntax of the AREA_INFORMATION_RECORD control statement, see “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61. Correct the error, and rerun the job.

FABC0715E	- INVALID AREA_INFORMATION_RECORD= CONTROL CARD - "AREA_INFORMATION_RECORD" CONTROL CARD PREVIOUSLY SPECIFIED
------------------	--

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

For details on the syntax of the AREA_INFORMATION_RECORD control statement, see “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61. Correct the error, and rerun the job.

FABC0717E	INVALID DBDNAME= CONTROL CARD - "DBDNAME=" CONTROL CARD PREVIOUSLY SPECIFIED
------------------	---

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61 for details about the syntax of the DBDNAME control statement. Correct the error, and rerun the job.

FABC0718E	INVALID DBDNAME= CONTROL CARD - DBD NAME SPECIFIED TOO LONG
------------------	--

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61 for details about the syntax of the DBDNAME control statement. Correct the error, and rerun the job.

FABC0720I	- SDEP=PHYSICAL UNLOADED SEGMENT RECORDS FOUND AND IGNORED
------------------	---

Explanation

FABCUR7 found SDEP=PHYSICAL unloaded segment records. They were ignored because they are not actual SDEP image segment records.

System action

Program FABCUR7 discards them and continues processing.

User response

Check if the input file is correct.

FABC0721I	OBTAINED DB DEFINITIONS FROM <i>resource</i>
------------------	---

Explanation

This message indicates the resource (ACB library or IMS directory) where FABCUR7 obtained database definitions from.

System action

FABCUR7 continues processing.

User response

None. This message is informational.

FABC0722E	INVALID IMSCATHLQ= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61. Correct the error and rerun the job.

FABC0723E	INVALID IMSCATHLQ= CONTROL CARD - "IMSCATHLQ=" CONTROL CARD PREVIOUSLY SPECIFIED
------------------	---

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61. Correct the error and rerun the job.

FABC0724E	INVALID IMSCATHLQ= CONTROL CARD - IMSCATHLQ SPECIFIED TOO LONG
------------------	---

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61. Correct the error and rerun the job.

FABC0725E	INVALID IMSCATACB_INPUT= CONTROL CARD - SYNTAX ERROR DETECTED
------------------	--

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61. Correct the error and rerun the job.

FABC0726E	INVALID IMSCATHLQ_INPUT= CONTROL CARD - "IMSCATHLQ_INPUT=" CONTROL CARD PREVIOUSLY SPECIFIED
------------------	---

Explanation

Self-explanatory.

System action

FABCUR7 ends with an abend code of 3728.

User response

See “Input for the DEDB Unloaded Segment Data Set Retrieve utility” on page 61. Correct the error and rerun the job.

FABC0800I	- FABCUR8 PROCESSING STARTED
------------------	-------------------------------------

Explanation

This message is generated when FABCUR8 starts the requested processing.

System action

Program FABCUR8 continues processing.

User response

None. This message is informational.

FABC0801I	- FABCUR8 ENDED NORMALLY
------------------	---------------------------------

Explanation

This message is generated when all requested processing have been completed without errors.

System action

Program FABCUR8 ended with a return code of 0.

User response

None. This message is informational.

FABC0801W - FABCUR8 ENDED WITH WARNINGS

Explanation

This message is generated when trivial error conditions were encountered by program FABCUR8.

System action

FABCUR8 ends with a return code of 4.

User response

To determine the nature and causes of the errors detected, see the other messages that were generated by FABCUR8. Correct the problem and rerun the job, or continue with reload processing, as desired.

FABC0801E - FABCUR8 ENDED WITH ERRORS

Explanation

This message is generated when nontrivial error conditions were encountered by program FABCUR8.

System action

FABCUR8 ends with a return code of 8.

User response

To determine the nature and causes of the errors detected, see the other messages that were generated by FABCUR8. Correct the problem, and rerun the job.

FABC0802I - HD DBD NAME IS NOT SPECIFIED IN THE EXEC PARAMETER

Explanation

Program FABCUR8 detected that the HD DBD name is not specified in the EXEC parameter.

System action

FABCUR8 continues processing.

User response

FABCUR8 cannot verify the fixed-length segment and the compressed segment by the HD DBD, so the result may be unpredictable if HD unload file contains fixed-length or compressed segments. Make sure that HD unload files does not contain fixed-length or compressed segments.

FABC0803W - THE HD UNLOAD FILE BELONGS TO HALDB DBD: *dbdname*

Explanation

Program FABCUR8 detected that the input HD unload file belongs to HALDB DBD specified by *dbdname*. To convert all records of the HALDB database to a DEDB database, HD unload files of all partitions of the HALDB must be specified.

System action

FABCUR8 sets a return code of 4 and continues.

User response

Make sure that HD unload files of all partitions of the HALDB database are concatenated as input. If the input is not for all partitions, then the FABCUR8 output unloaded segment records file must be sorted or merged with the output of another partitions before they are reloaded to the DEDB by using FABCUR3.

FABC0810E - SEGMENT: *segname* IN HD UNLOAD FILE IS NOT DEFINED IN HD DBD: *dbdname*

Explanation

The segment specified by *segname* in the HD unload file is not defined in the HD DBD specified by *dbdname*.

System action

Program FABCUR8 ends with a return code of 8.

User response

Check the reason of the error, specify the correct HD unload file or the correct DBD, and rerun the job.

FABC0811E - SEGMENT CODE OF SEGMENT: *segname* BETWEEN HD UNLOAD FILE AND HD DBD: *dbdname* DOES NOT MATCH
**- HD UNLOAD FILE: *nnn* (X'*xx*')
- HD DBD: *nnn* (X'*xx*')

---****Explanation**

The segment code of the segment specified by *segname* in the HD unload file and the HD DBD specified by *dbdname* did not match.

System action

Program FABCUR8 ends with a return code of 8.

User response

Check the reason of the error, specify the correct HD unload file or the correct DBD, and rerun the job.

FABC0812E - **SEGMENT LEVEL OF SEGMENT: *segname* BETWEEN HD UNLOAD FILE AND HD DBD: *dbdname* DOES NOT MATCH**
- **HD UNLOAD FILE: *nn* (X'*xx*')**
- **HD DBD: *nn* (X'*xx*')**

Explanation

The segment level of the segment specified by *segname* in the HD unload file and the HD DBD specified by *dbdname* did not match.

System action

Program FABCUR8 ends with a return code of 8.

User response

Check the reason of the error, specify the correct HD unload file or the correct DBD, and rerun the job.

FABC0813E - **SEGMENT: *segname* IN HD UNLOAD FILE DEFINED IN HD DBD: *dbdname* IS A FIXED-LENGTH SEGMENT**

Explanation

The segment specified by *segname* in the HD unload file that is defined in the HD DBD specified by *dbdname* is a fixed-length segment. Program FABCUR8 does not support fixed-length segment for input.

System action

FABCUR8 ends with a return code of 8.

User response

If you specify the incorrect HD unload file or incorrect DBD, specify the correct HD unload file or the correct DBD, and rerun the job.

FABC0814E - **SEGMENT: *segname* IN HD UNLOAD FILE DEFINED IN HD DBD: *dbdname* IS A COMPRESSED SEGMENT**

Explanation

The segment specified by *segname* in the HD unload file defined in the HD DBD specified by *dbdname* is

a compressed segment. Program FABCUR8 does not support compressed segment for input.

System action

FABCUR8 ends with a return code of 8.

User response

If you had specified an incorrect HD unload file or an incorrect DBD, specify the correct HD unload file or the correct DBD, and rerun the job.

FABC0815E - **FIRST RECORD OF HD UNLOAD FILE IS NOT A HEADER RECORD**

Explanation

Program FABCUR8 found that the first record of the HD unload file that is specified by the DURINPT DD statement was not an HD unload header record.

System action

FABCUR8 ends with a return code of 8.

User response

Specify the correct HD unload file, and rerun the job.

FABC0816E - **HD UNLOAD FILE SPECIFIED BY DURINPT DD IS EMPTY**

Explanation

Program FABCUR8 found that the HD unload file that is specified by the DURINPT DD statement was empty.

System action

FABCUR8 ends with a return code of 8.

User response

Specify the correct HD unload file, and rerun the job.

FABC0817E - **SEGMENT: *segname* IN HD UNLOAD FILE IS A FIXED-LENGTH SEGMENT**

Explanation

The HD DBD name is not specified in the EXEC parameter but FABCUR8 found that the segment that is specified by *segname* in the HD unload file is a fixed-length segment. The first two bytes of the segment data is not a LL value. Program FABCUR8 does not support fixed-length segment for input.

System action

FABCUR8 ends with a return code of 8.

User response

Specify the correct HD unload file or the correct DBD, and rerun the job.

FABC0818E - SEGMENT: *segname* DEFINED IN HD DBD: *dbdname* IS NOT FOUND IN THE HD UNLOAD HEADER RECORD

Explanation

The segment that is specified by *segname* defined in the HD DBD specified by *dbdname* is not found in the HD unload header record entry. Because this defines as a physical segment, the HD unload header record must have an entry for the segment.

System action

Program FABCUR8 ends with a return code of 8.

User response

Specify the correct HD unload file or the correct DBD, and rerun the job.

FABC0820E - SEGMENT: *segname* IN HD UNLOAD FILE NOT DEFINED IN DEDB DMB: *dmbname*

Explanation

The segment specified by *segname* in the HD unload file is not defined in the DEDB DMB specified by *dmbname*.

System action

Program FABCUR8 ends with a return code of 8.

User response

Check the reason of the error, specify the correct HD unload file or the correct DEDB DMB, and rerun the job.

FABC0821E - PARENT OF SEGMENT: *segname* IN HD UNLOAD FILE AND DEDB DMB: *dmbname* DOES NOT MATCH
- HD UNLOAD FILE: *parent-segname*
- DEDB DMB: *parent-segname*

Explanation

Parent segment name of the segment specified by *segname* in the HD unload file and the DEDB DMB specified by *dmbname* did not match.

System action

Program FABCUR8 ends with a return code of 8.

User response

Check the reason of the error, specify the correct HD unload file or the correct DEDB DMB, and rerun the job.

FABC0822E - HIERARCHY OF SEGMENT: *segname* UNDER THE PARENT IN HD UNLOAD FILE AND DEDB DMB: *dmbname* DOES NOT MATCH
- PARENT: *parent-segname*

Explanation

Hierarchy of the segment that is specified by *segname* under the parent specified by *parent-segname* in the HD unload file and that under the DEDB DMB that is specified by *dmbname* did not match.

System action

Program FABCUR8 ends with a return code of 8.

User response

Check the reason of the error, specify the correct HD unload file or the correct DEDB DMB, and rerun the job.

FABC0823E - ROOT SEGMENT: *segname* IN HD UNLOAD FILE IS NOT A ROOT IN DEDB DMB: *dmbname*

Explanation

The root segment specified by *segname* in the HD unload file is not a root in the DEDB DMB specified by *dmbname*.

System action

Program FABCUR8 ends with a return code of 8.

User response

Check the reason of the error, specify the correct HD unload file or the correct DEDB DMB, and rerun the job.

FABC0824E - SEGMENT: *segname* DEFINED IN DEDB DMB: *dmbname* IS A FIXED-LENGTH SEGMENT

Explanation

The segment specified by *segname* defined in the DEDB DMB specified by *dmbname* is a fixed-length segment. Program FABCUR8 does not support fixed-length segment for output.

System action

FABCUR8 ends with a return code of 8.

User response

If you had specified an incorrect DEDB DMB, specify the correct DEDB DMB, and rerun the job.

FABC0900I ddname: <text>

Explanation

The text displays the data contained on a record read from the file.

DDNAME is one of the following:

- CNTLCRDS
- SEGXREFI

System action

Processing continues.

User response

None. This message is informational.

Module

FABCUR9

FABC0901E NON-NUMERIC DATA IN <keyword> FIELD

Explanation

The value associated with a keyword should be numeric; however, non-numeric data was found.

System action

Processing ends with return code 8. This message will be accompanied by message FABC0909E, which will display the control statement in error.

User response

Correct the statement in error, and resubmit job.

Module

FABCUR9P

FABC0902E SEGXREFI FILE IS NOT IN SEQUENTIAL, CONTIGUOUS ORDER AT RECORD # xxxxxxxx

Explanation

The Segment Code variable contained in the records in this file must be in sequential ascending order. The message identifies the record which was detected to be out of order.

System action

Processing ends with return code 08.

User response

Correct the order of statements in the file and resubmit job.

Module

FABCUR9

FABC0903E <file name> FILE IS REQUIRED, BUT IS EMPTY OR INVALID

Explanation

The invalid file may be one of the following:

- DURDBDFN

The DURDBDFN DD statement was present, but the file was empty.

- SEGXREFI

The CNTLCRDS file contained the keyword 'SEGXREFI', which specifies that records are to be processed from this file. The SEGXREFI file was found to be empty.

System action

Processing ends with return code 8.

User response

Take the following actions:

- DURDBDFN

Make sure that a file containing a valid database definition record is referenced by the DURDBDFN DD statement.

- **SEGXREFI**

Make sure that the correct records are in the file, or remove the SEGXREFI statement from the CNTLCRDS file and resubmit job.

Module

FABCUR9

FABC0904E	INVALID SEG CODE <segcode> ENCOUNTERED AT RECORD # xxxxxxx COLUMN # xxxxxx
------------------	---

Explanation

The UNLDREC file contained a record containing the specified segment code. This segment code was not determined to be valid for the database being processed.

System action

Processing ends with return code 8.

User response

Determine and correct the problem. One of the following may be in error:

- The DURDBDFN file may not match the database being processed, if this file is being used.
- The DURDBDFN file may not have been intended to be used; however, the file is not empty.
- The SEGXREFI file may not match the database being processed, if this file is being used.
- The SEGXREFI file may not have been intended to be used; however, a CNTLCRDS record specifies 'SEGXREFI'
- The UNLDREC file may contain an unsupported record format

Module

FABCUR9F

FABC0905E	RECORD # : xxxxxxxx SEG CODE : xxx KEY LEN : xxxxxxxx APPEARS TO BE AN INVALID UNLOAD FORMAT RECORD
------------------	--

Explanation

The UNLDREC file contained a record which FABCUR9 was not able to interpret.

System action

Processing ends with return code 8.

User response

Determine and correct the problem. One of the following may be in error:

- The UNLDREC file may contain an unsupported record format.

Module

FABCUR9F

FABC0906E	PROBLEM RESOLVING INPUT <input>
------------------	--

Explanation

A CNTLCRDS record specified an invalid INPUT value.

System action

Processing terminates with return code 8.

User response

Correct the invalid keyword and resubmit the job.

Module

FABCUR9

FABC0907E	DBDNAME= PARAMETER HAS NOT BEEN PROVIDED
------------------	---

Explanation

A value for DBDNAME was required; however, no control statement containing this value was supplied.

System action

Processing ends with return code 8.

User response

Include the DBDNAME= keyword in a CNTLCRDS control statement, and resubmit the job.

Module

FABCUR9

FABC0908E	IF ANY DURDBDFN COMPONENTS ARE SPECIFIED, ALL MUST BE
------------------	--

Explanation

There are three keywords related to overriding the default values for lengths of DURDBDFN components. Keywords were included in the CNTLCRDS file to override one or more of these values; however, at least one keyword was omitted.

System action

Processing ends with return code 8.

User response

Determine and correct the problem using one of the following procedures:

- Include all three parameters in the CNTLCRDS file. Consult the manual for the proper syntax.
- Remove any of the three parameters which are in the CNTLCRDS file.

Module

FABCUR9

FABC0909E INVALID ddname: <text>

Explanation

The specified record contained an invalid statement. DDNAME is one of the following:

- CNTLCRDS
- SEGXREFI

System action

Processing ends with return code 8.

User response

Correct the statement, and resubmit job.

Module

FABCUR9 FABCUR9P

**FABC0913I DURDBDFN COMPONENT
LENGTHS SUPPLIED IN CNTLCRDS**

Explanation

The default values for the lengths of the three DURDBDFN components was overridden by values contained in the CNTLCRDS file.

System action

Processing continues.

User response

None. This message is informational.

Module

FABCUR9

**FABC0914E INVALID INPUT <input> HAS BEEN
SPECIFIED**

Explanation

A CNTLCRDS record specified an invalid INPUT value.

System action

Processing terminates with return code 8.

User response

Correct the invalid keyword and resubmit the job.

Module

FABCUR9

**FABC0915E THE REQUESTED DBDNAME:
<dbdname> IS NOT REFERENCED
IN THE PSB**

Explanation

The specified DBDNAME did not correspond to a PCB in the PSB.

System action

Processing ends with return code 8.

User response

Determine and correct the problem. One of the following errors may exist:

- The DBDNAME specified in the CNTLCRDS DBDNAME= statement does not correspond to a PCB in the PSB.
- The DBDLIB DD statement references a DBD which does not correspond to a PCB in the PSB.
- An incorrect PSB may be specified in the EXEC statement of DFSRRC00
- The intent may have been to run program FABCUR9 in test mode, which does not validate the DBD name

or issue DL/I calls; however, the 'TEST' keyword was not included in CNTLCRDS

Module

FABCUR9

FABC0916E	CHECKPOINT RESTART PROCESSING IS NOT SUPPORTED
------------------	---

Explanation

An attempt was made to checkpoint restart.

System action

Processing ends with return code 8.

User response

Do not attempt the check point restart. In many cases, such a restart is not required. Consult the manual for Recovery restrictions.

Module

FABCUR9

FABC0917E	UNABLE TO DETERMINE LANGUAGE OF PSB
------------------	--

Explanation

The utility was unable to determine the type of PSB which was in use.

System action

Processing ends with return code 8.

User response

This problem may be an internal error in the FABCUR9 Utility. Contact IBM Software Support.

Module

FABCUR9

FABC0918E	SEGXREFI FIXED FORMAT AREA FIELDS ARE IN INCORRECT COLUMNS
------------------	---

Explanation

Some of the fields within the first 43 positions of the fixed format area of the SEGXREFI input field are misaligned or in the wrong columns.

System action

Processing ends with return code 8.

User response

Correct the SEGXREFI file. Consult the user's guide for the proper position of fields within the segment cross-reference file records.

Module

FABCUR9

FABC0919W	RECORD FORMAT <format> INCORRECT FOR SEGMENT <segname>
------------------	---

Explanation

A DL/I call returned a status code of 'LD' because the implied or specified record format ('V' or 'F') for the I/O area of the specified segment did not match the DBD specification for this segment.

System action

Processing continues.

The utility will use the alternative record format and will reattempt the DL/I call.

User response

If the Segment Cross-Reference Records are being used, the RECORD FORMAT value for this segment should be changed from 'V' to 'F' or vice versa.

Module

FABCUR9I

FABC0920E	SEGXREFI FILE WITH VALID RECORD FORMAT VALUES REQUIRED FOR INPUT=<input> PROCESSING
------------------	--

Explanation

A value for record format is required for each segment in the Segment Cross-Reference File (SEGXREFI) for this unload file format type

The following INPUT types require RECORD FORMAT values:

- IMS High Performance Unload/Reload Format Records (FF)

System action

Processing terminates with return code 8.

User response

Create Segment Cross-Reference records which include the RECORD FORMAT value for each segment and resubmit the job.

Module

FABCUR9F

FABC0921E	SEGXREFI RECORD FOR SEGMENT <i>segname</i> CONTAINS MISSING OR INVALID DATA FOR FIELD <i>fieldname</i>
------------------	---

Explanation

The Segment Cross-Reference record for the specify segment was missing data for the specified field.

The specified fields are among the following:

- Segment ID
- Segment Level
- Parent
- Segment name
- Position (Blank or Zero is acceptable)
- Maximum Length (Blank or Zero is acceptable)
- Minimum Length (Blank or Zero is acceptable)

System action

Processing ends with return code 8.

User response

Specify values for the omitted fields and resubmit the job.

Module

FABCUR9

FABC0922E	UNSUPPORTED DLI CALL: <i>calltype</i> ISSUED
------------------	---

Explanation

The specified call type is not supported by the program.

System action

Processing will abend with code 3401.

User response

This problem may be an internal error in the FABCUR9 Utility. Contact IBM Software Support.

Module

FABCUR9I

FABC0923E	OPEN ERROR ON <i>dsname</i> DATASET
------------------	--

Explanation

The specified data set could not be opened.

System action

Processing ends with return code 8. This message will

User response

Supply a DD statement for the specified data set, or remove the request for the data set from the control cards. FABCUR9 Utility. Contact IBM Software Support.

Module

FABCUR9

FABC0924E	UNABLE TO VALIDATE SEG CODE <i>xxx</i> WITH CODE <i>xxx</i> ENCOUNTERED AT RECORD # <i>xxxxxxx</i> COLUMN # <i>xxxxxxx</i>
------------------	---

Explanation

A problem was encountered while determining the correct segment code for the record.

System action

Processing ends with return code 8.

User response

This problem may be an internal error in the FABCUR9 Utility. Contact IBM Software Support.

Module

FABCUR9F

FABC0925E	MUTUALLY EXCLUSIVE SEG CROSS REF RECORDS ENCOUNTERED FOR SEGMENT <i>segname</i>
------------------	--

Explanation

Mutually exclusive keywords were encountered in the Segment Cross-Reference File (SEGXREFI) for the specified segment.

The following combinations are not allowed:

- REPL and NOREP
- REPL and GHU
- BYPASS and REPL
- BYPASS and GHU

System action

Processing ends with return code 8.

User response

Resolve the conflicting parameters, and resubmit the job.

Module

FABCUR9

FABC0926E	LOAD OF <i>segname</i> ATTEMPTED INTO A POPULATED DATABASE.
------------------	--

Explanation

An attempt was made to perform updates in a database which was already populated. The REPL CNTLCRDS keyword was not specified

System action

Processing ends with return code 8.

User response

Determine whether load processing was intended

- If load processing was intended, scratch and reallocate the database
- If updates to the existing database were intended, specify the REPL CNTLCRDS keyword

Correct the situation, and resubmit the job

Module

FABCUR9U

FABC0927E	BYPASS FOR SEGMENT <i>segname1</i> CONFLICTS WITH INTENT FOR SEGMENT <i>segname2</i>
------------------	---

Explanation

The segment identified by *segname2* may be processed; however, the Segment Cross-Reference table record for a segment higher in the hierarchy, identified by *segname2*, contains the BYPASS parameter. It is illegal for a dependent segment to be processed if a segment higher in the hierarchy is to be bypassed.

System action

Processing ends with return code 8.

User response

Change the designation of one of the involved segments:

- Add the BYPASS parameter to the dependent segment if processing is not required
- Change the parameter for the parent segment from BYPASS to NOREP if processing of the parent is not required.

Correct the situation and resubmit the job

Module

FABCUR9

FABC0929E	REPL/GHU INTENT FOR SEGMENT <i><segname></i> NOT ALLOWED WITH NON-KEYED PARENT <i><parent-segname></i>
------------------	---

Explanation

The segment identified by *segname* has a non-keyed parent in the hierarchical chain. Keywords REPL or GHU were specified for this segment, but replace-related processes are not supported for segments where a parent does not have a key.

System action

Processing ends with return code 8.

User response

Specify BYPASS in the segment cross-reference table for these segments.

Correct the situation and resubmit the job

Module

FABCUR9

FABC0930E **REPL/GHU INTENT NOT ALLOWED
FOR NON-KEYED SEGMENT**
segname

Explanation

The segment identified by *segname* is a non-keyed segment. Keywords REPL or GHU were specified for this segment, but replace-related processes are not supported for segments without keys.

System action

Processing ends with return code 8.

User response

Specify BYPASS in the segment cross-reference table for these segments.

Correct the situation and resubmit the job

Module

FABCUR9

FABC0931E **STORAGE OBTAIN FAILURE ON
AREA: *areaname***

Explanation

A GETMAIN of a storage area was attempted, but was unsuccessful.

System action

Processing ends with return code 8. This message will

User response

This problem may be an internal error in the FABCUR9 Utility. Contact IBM Software Support.

Module

FABCUR9

FABC0932E **REPL/GHU INTENT NOT ALLOWED
FOR UNKNOWN KEY STATUS
SEGMENT *segment***

Explanation

The segment cross-reference file does not contain information describing whether the segment is keyed, non-keyed or an SDEP. Keywords REPL or GHU were specified for this segment but replace-related processes are not supported for segments where the key status is not known. If the segment cross-

reference file was created during processing of a Full Function unload file, the unload file does not contain the key status information. In this situation, the key status will be initialized to hex zeros in the segment cross-reference file, and must be updated correctly before this file can be used to drive replace-related processes

System action

Processing ends with return code 8.

User response

Correct the key status information in the segment cross-reference table and resubmit the job.

Module

FABCUR9

FABC0933E **INVALID VALUE: <value> FOR
FIELD: <field name> IN FILE <file
name>**

Explanation

The specified input file contains an invalid value for the specified field.

System action

Processing ends with return code 8.

User response

Determine the correct value for the field, and correct the input.

Module

FABCUR9

FABC0934E **UNLOADED DATASET CONTAINING
COMPRESSED SEGMENTS IS NOT
SUPPORTED**

Explanation

The DEDB unloaded data set specified by the UNLDREC DD contained compressed unloaded segments.

System action

Processing ends with return code 8.

User response

Unload data set containing decompressed segments, and resubmit the job.

Module

FABCUR9F

FABC0935E	UNLOADED DATASET CONTAINING SEGMENTS REACHED INSERT LIMIT COUNT IS NOT SUPPORTED
------------------	---

Explanation

The DEDB unloaded data set specified by the UNLDREC DD contained segments which reached the ILC criteria.

System action

Processing ends with return code 8.

User response

Unload data set without ILC option, and resubmit the job.

Module

FABCUR9F

FABC0936I	UNLOADED DATASET CONTAINED SEGMENTS WITH SUBSET POINTER(S) INFORMATION
------------------	---

Explanation

The DEDB unloaded data set specified by the UNLDREC DD contained segments that contained subset pointer information. FABCUR9 Utility did not load subset pointer for the segment.

System action

Processing continues. The subset pointer for any segments with subset pointers will not be loaded.

User response

No action is required.

Module

FABCUR9F

FABC0937E	UNSUPPORTED UNLOADED DATA SET DETECTED
------------------	---

Explanation

The unload data set specified by the UNLDREC DD statement was in a format not supported by FABCUR9 Utility.

System action

Processing ends with return code 8.

User response

Make sure that only data sets in supported unload formats are specified. Consult the user's guide for a description of supported formats. Correct the errors, and resubmit the job.

Module

FABCUR9F

FABC0938I	DEDB SDEP SEGMENTS WITH SDEP=PHYSICAL FORMAT DETECTED AND IGNORED
------------------	--

Explanation

The DEDB unloaded data set specified by the UNLDREC DD contained segments with SDEP=PHYSICAL format. FABCUR9 Utility did not load SDEP physical records.

System action

FABCUR9 ignores all of SDEP segment records with SDEP=PHYSICAL format and continues processing.

User response

No action is required.

Module

FABCUR9F

FABC3700E	>>>>> UNLOAD FAILED FOR AREA nnnnn (AREANAME: areaname) - ERROR OCCURRED DURING CALL TO RANDOMIZER rmodname FUNC: RANDOMIZING CALL
------------------	--

Explanation

Program FABCUR1 determined that the unload subtask processing the specified area was unable to complete successfully. The first message is issued when an abend occurs during a call to the database Randomizer module. "FUNC: RANDOMIZING CALL"

is issued only when the unload subtask processing invoked the XCI randomizer.

System action

FABCUR1 ends with an abend code of 3700.

User response

Examine the other FAB37xx messages generated to determine the nature of the problem. Correct the problem, and rerun the job.

FAB3703E	MEDIA MANAGER I/O ERROR - AREA zzzz9 (DDNAME ddname) - REQUESTED RBA: eeeeeeee - MEDIA MANAGER RETURN CODE: ccccffss
-----------------	---

Explanation

When an unload subtask issued the MMGRCALL to get access to the data set associated with the *ddname* specified, an unexpected Media Manager MMGRCALL error occurred. The variable *cccccffss* represents the Media Manager error return code used for problem determination. Media Manager return codes are described in the *DFSMS: DFSMSdfp Diagnosis Reference*. *zzzz9* is the area number.

System action

The unload subtask ends with an abend code of 3703.

User response

Check the Media Manager return code, correct the error, and rerun the job. If the error persists, contact IBM Software Support for additional analysis.

FAB3704E	MEDIA MANAGER CONNECT DISCONNECT ERROR - AREA zzzz9 (DDNAME ddname) - MEDIA MANAGER RETURN CODE: ccccffss
-----------------	---

Explanation

When an unload program issued the MMGRSRV to connect or disconnect the data set associated with the *ddname* specified, an unexpected Media Manager MMGRSRV error occurred.

The variable *cccccffss* represents the Media Manager error return code that can be used for problem determination. Media Manager return codes are described in the *DFSMS: DFSMSdfp Diagnosis Reference*. *zzzz9* is the area number.

System action

The unload subtask ends with the abend code of 3704.

User response

Check the Media Manager return code, correct the error, and rerun the job. If the error persists, contact IBM Software Support for additional analysis.

FAB3705E	INSUFFICIENT STORAGE: INCREASE REGION SIZE (aaaa)
-----------------	--

Explanation

Programs FABCUR1/FABCUR5/FABCUR6/FABCUR7/FABCUR8 issued a GETMAIN macro to allocate storage for the purpose of *aaaa*. The attempt was unsuccessful.

System action

FABCUR1/FABCUR5/FABCUR6/FABCUR7/FABCUR8 ends with an abend code of 3705.

User response

Check the unload region size. Increase the REGION parameter on the EXEC statement for FABCUR1/FABCUR5/FABCUR6/FABCUR7/FABCUR8 as required. Rerun the job.

FAB3706E	ERROR ATTACHING UNLOAD SUBTASK (RC = xx)
-----------------	---

Explanation

Program FABCUR1 issued an SVC 42 (ATTACH) to activate an unload subtask. The return code from OS specified that the attempt was unsuccessful.

System action

FABCUR1 ends with an abend code of 3706.

User response

For further information, see the *MVS Programming: Assembler Services Reference*. Correct any errors, and rerun the job.

FAB3707E	CORRECT IMS RESLIB NOT CONCATENATED [- NO DFSBSCDO MODULE FOUND] [- INVALID IMS LEVEL]
-----------------	---

Explanation

Correct IMS load module library was not concatenated to the JOBLIB/STEPLIB because JOBLIB/STEPLIB library has no DFSBSCD0 module or DFSBSCD0 module shows unsupported IMS level.

System action

Program FABCUR1/FABCUR3 ends with an abend code of 3707.

User response

Concatenate the correct IMS load module library to the JOBLIB/STEPLIB, and rerun the job.

FABC3708E	IMS ONLINE SYSTEM IS ACCESSING AREA zzzzz (AREANAME: areaname) - WITH UPDATE/EXCLUSIVE INTENT
------------------	--

Explanation

Program FABCUR1 found that the area specified was being used by an IMS online system with update intent.

System action

FABCUR1 ends with an abend code of 3708.

User response

Stop the area on the IMS online system(s) by entering / STOP AREA or /DBR AREA command or change the access intent of the area to read intent on the IMS system(s), and rerun the job if desired.

FABC3710E	"OPEN" FAILED FOR DDNAME ddname - FAILED BY OS - DD STATEMENT NOT FOUND OR DUMMY/NULLFILE SPECIFIED - NOT A FIXED LENGTH RECORD DATASET - NOT A VARIABLE LENGTH RECORD DATASET - NOT AN 80 BYTE RECORD DATASET - RECORD LENGTH (LRECL) TOO SMALL (xxxxx REQUIRED) (xxxxx SPECIFIED) - BLOCK SIZE (BLKSIZE) TOO SMALL (xxxxx REQUIRED) (xxxxx SPECIFIED)
------------------	--

Explanation

Program FABCUR1/FABCUR5/FABCUR6/FABCUR8 issued an OPEN for the ddname specified. The OPEN failed.

System action

FABCUR1/FABCUR5/FABCUR6/FABCUR8 ends with an abend code of 3710.

User response

Make sure that a DD statement is present for the ddname specified, and that it properly identifies the correct data set. Correct any errors, and rerun the job.

FABC3711E	IMS TOOLS CATALOG INTERFACE <i>function</i> FUNCTION (DEFINITION=[CURRENT PENDING] FAILED - RETURN CODE: rc, REASON CODE: rsn
------------------	--

Explanation

The IMS Tools Catalog Interface ended with an error. *function* shows the function code of the IMS Tools Catalog Interface. The return code and reason code from the IMS Tools Catalog Interface are shown in *rc* and *rsn*, respectively.

System action

FABCUR6 or FABCUR7 ends with an abend code of U3711.

User response

If the function is OPEN, check if the correct high-level qualifier of the bootstrap data set is specified in the IMSCATHLQ keyword. Otherwise, contact IBM Software Support.

FABC3712E	MEMBER <i>acbname</i> NOT FOUND IN IMS CATALOG
------------------	---

Explanation

Program FABCUR6/FABCUR7 called an internal routine to obtain DMB information from the IMS directory. The return code from the routine indicates that the member does not exist.

System action

FABCUR6 or FABCUR7 ends with an abend code of 3712.

User response

Ensure that the high-level qualifier of the bootstrap data set of the IMS directory is correctly specified in the IMSCATHLQ keyword. Also, ensure that the IMS catalog population was correctly performed for the

database being processed. Correct any errors, and rerun the job.

FABC3713E **MEMBER *member_name* FROM
IMS CATALOG DEFINES DATABASE
*dbdname***

Explanation

Program FABCUR6/FABCUR7 called an internal routine to obtain DMB information from the IMS directory. The return code from the routine indicates that the name of the DEDB DMB member does not match the name of the database.

System action

FABCUR6 or FABCUR7 ends with an abend code of 3713.

User response

Ensure that the high-level qualifier of the bootstrap data set of the IMS directory is correctly specified in the IMSCATHLQ keyword. Also, ensure that the IMS catalog population was correctly performed for the database being processed. Correct any errors, and rerun the job.

FABC3714E **MEMBER *member_name* FROM IMS
CATALOG IS NOT A DEDB DMB**

Explanation

Self-explanatory.

System action

FABCUR6 or FABCUR7 ends with an abend code of 3714.

User response

Ensure that the high-level qualifier of the bootstrap data set of the IMS directory is correctly specified in the IMSCATHLQ keyword. Also, ensure that the IMS catalog population was correctly performed for the database being processed. Correct any errors, and rerun the job.

FABC3715E **IMS LEVEL OF MEMBER
member_name FROM IMS
CATALOG IS NOT SUPPORTED**

Explanation

Self-explanatory.

System action

FABCUR6 or FABCUR7 ends with an abend code of 3715.

User response

Ensure that the high-level qualifier of the bootstrap data set of the IMS directory is correctly specified in the IMSCATHLQ keyword. Also, ensure that the IMS catalog population was correctly performed for the database being processed. Correct any errors, and rerun the job.

FABC3719E **OPEN FAILED FOR VSAM DATASET
AT DDNAME *ddname* (AREA *zzzzz*)
- VSAM ERROR DATA: RETURN-
CODE: *rrr*
ACB "ERROR": *aaa* (*bb*)
- DD STATEMENT NOT FOUND**

Explanation

Program FABCUR1 issued an OPEN for the *ddname* specified. The return code (*rrr*) from VSAM specified that the OPEN failed. The error code value from the ACB is shown in both decimal (*aaa*) and hexadecimal (*bb*) formats. *zzzzz* is the area number.

System action

FABCUR1 ends with an abend code of 3719.

User response

Make sure that an appropriate DD statement is present for the specified *ddname* and that it properly identifies the correct data set. See *DFSMS Macro Instructions for Data Sets*, which describes VSAM administration macros.

FABC3720E **"DEVTYPE" FAILED FOR DDNAME
ddname (RC=*xx*)**

Explanation

Program FABCUR1/FABCUR5/FABCUR6/FABCUR7/FABCUR8 issued an SVC 24 (DEVTYPE) to obtain information about the input/output device associated with *ddname*. The return code shows that the attempt was unsuccessful.

System action

FABCUR1/FABCUR5/FABCUR6/FABCUR7/FABCUR8 ends with an abend code of 3720.

User response

Correct any errors, and rerun the job. If this situation persists, report it to systems operations personnel.

FABC3721E	CALL TO "GET DEDB DMB" ROUTINE FAILED (RC = zz)
------------------	--

Explanation

Program FABCUR1/FABCUR3/FABCUR5/FABCUR6 called an internal routine to read and analyze a member from the ACB library data set. The return code from the routine (as shown in the message) shows that the attempt was unsuccessful.

System action

FABCUR1/FABCUR3/FABCUR5/FABCUR6 ends with an abend code of 3721.

User response

Contact IBM Software Support.

FABC3722E	TRKCALC FAILED FOR DDNAME ddname (RC = zz)
------------------	---

Explanation

Program FABCUR1 invoked the TRKCALC macro to determine the 'number of CIs per track' value for the device on which the data set associated with the specified *ddname* resides. The return code from OS specified that the attempt was unsuccessful.

System action

FABCUR1 ends with an abend code of 3722.

User response

For further information, see *DFSMS DFSMSdfp Advanced Services*. Correct the errors, and rerun the job. If this situation persists, contact IBM software Support.

FABC3723E	LOAD FAILED FOR [COMPRESSION ROUTINE EXIT ROUTINE RANDOMIZER ROUTINE] xxxxxxxx (ABEND CODE Sxxxx / REASON CODE xxxxxxxx)
------------------	---

Explanation

Program FABCUR1/FABCUR3/FABCUR6/FABCUR7 issued an SVC 8 (LOAD) to bring a copy of the randomizer routine or segment edit/compression

routine into the core. The return code from OS ('Abend Code') specifies that the attempt was unsuccessful.

System action

FABCUR1/FABCUR3/FABCUR6/FABCUR7 ends with an abend code of 3723.

User response

For further information, see the *MVS Programming: Assembler Services Reference*. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

FABC3724E	macro-name FAILED FOR DMB MEMBER member-name FOR DDNAME ddname (RC = rr; REASON = zz)
------------------	--

Explanation

Program FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 issued the specified macro (*macro-name*) to get access to the specified DMB member (*member-name*) in the data set specified (*ddname*). The return code and the reason code for z/OS specify that the attempt was unsuccessful.

System action

FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 ends with an abend code of 3724.

User response

For further information, see the *MVS Programming: Assembler Services Reference*. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

FABC3725E	CONTROL CARD DATASET IS EMPTY
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 ends with an abend code of 3725.

User response

The control statement stream must contain one DBDNAME statement and at least one AREACTL statement. Correct the control statement stream, and rerun the job.

FABC3726E NO VALID DBDNAME= SPECIFICATION FOUND

Explanation

Self-explanatory.

System action

Program FABCUR1 ends with an abend code of 3726.

User response

The control statement stream must contain one DBDNAME statement, and it must be the first control statement. Correct the control statement stream, and rerun the job.

FABC3727E NO VALID AREACTL= SPECIFICATION FOUND

Explanation

Self-explanatory.

System action

Program FABCUR1 ends with an abend code of 3727.

User response

The control statement stream must contain at least one AREACTL statement. Correct the control statement stream, and rerun the job.

FABC3728E SEVERE CONTROL CARD ERROR(S) ENCOUNTERED

Explanation

Program FABCUR1/FABCUR5/FABCUR6/FABCUR7 detected one or more severe errors during the analysis of the control statement.

System action

FABCUR1/FABCUR5/FABCUR6/FABCUR7 ends with an abend code of 3728.

User response

Examine the FABC01xx/FABC05xx/FABC06xx/FABC07xx messages generated to determine the nature of the problem(s). Correct the errors, and rerun the job.

FABC3729E DMB MEMBER *member-name* FOR xxxxxxxxxxxx HAS NO DATA

Explanation

Program FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 tried to read all data of the DMB member (*member-name*) but the member has no data.

System action

FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 ends with an abend code of 3729.

User response

Make sure that the DD statement (ddname) properly specifies the correct data set. Correct the error, and rerun the job.

FABC3730E MEMBER *acbname* NOT FOUND IN *acb-ddname* DATASET

Explanation

Program FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 called an internal routine to read and analyze a DMB in the specified ACBLIB data set. The return code from the routine indicates that the member does not exist.

System action

FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 ends with an abend code of 3730.

User response

Make sure that the DD statement specified properly specifies the correct data set. If the ddname specified is NEWACB, make sure that the required DMB is present or remove the DD statement from the JCL stream. Correct the errors, and rerun the job.

FABC3731E MEMBER *member_name* FROM xxxxxxxxxxxx DEFINES DATABASE *dbdname*

Explanation

Program FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 called an internal routine to read and analyze a DMB in the specified ACBLIB data set or the IMS directory. The return code from the routine indicates that the name of the DEDB DMB member does not match the name of the database.

System action

FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 ends with an abend code of 3731.

User response

Make sure that the DD statement specified properly specifies the correct data set, and that the DBDGEN and ACBGEN, or IMS catalog population, have completed successfully for the database being processed. Correct the errors, and rerun the job.

FABC3732E	MEMBER <i>member_name</i> FROM xxxxxxxx IS NOT A DEDB DMB
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 ends with an abend code of 3732.

User response

Make sure that the DD statement specifies the correct data set, and that the DBDGEN and ACBGEN, or IMS catalog population, have completed successfully for the database being processed. Correct the errors, and rerun the job.

FABC3733E	AREA <i>xxxxx</i> SPECIFIED ON AREACTL= CARD NOT DEFINED IN DMB xxxxxxxx FROM xxxxxxxx
------------------	---

Explanation

Self-explanatory. xxxxx is the area number.

System action

Program FABCUR1 ends with an abend code of 3733.

User response

Verify the correctness of the user-supplied AREACTL control statements. Make sure that the OLDACB DD statement or the IMS directory is specified correctly. Correct any errors, and rerun the job.

FABC3734E	AREA <i>xxxxx</i> SPECIFIED ON FILECTL= CARD NOT DEFINED IN DMB xxxxxxxx FROM yyyyyyyy DATASET
------------------	---

Explanation

Self-explanatory. xxxxx is the area number.

System action

Program FABCUR1/FABCUR6 ends with an abend code of 3734.

User response

Verify the correctness of the user-supplied FILECTL control statements. Make sure that the DD statement specified specifies the correct data set. If the ddname specified is NEWACB, make sure that the required DBDGEN and ACBGEN have been correctly performed. Correct the errors, and rerun the job.

FABC3735E	SEGMENT <i>segname</i> SPECIFIED ON LOADCTL= CARD NOT DEFINED IN "OLD" DMB
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1 ends with an abend code of 3735.

User response

Verify the correctness of the user-supplied LOADCTL control statements. Make sure that the OLDACB DD statement specifies the correct data set. Correct the errors, and rerun the job.

FABC3736E	SEGMENT <i>segname</i> DEFINED IN "OLD" DMB; NOT FOUND IN "NEW" DMB
------------------	--

Explanation

Self-explanatory.

System action

Program FABCUR1 ends with an abend code of 3736.

User response

Verify that the OLDACB and NEWACB DD statements are specified correctly and the required DBDGEN and ACBGEN have been performed correctly. Correct the errors, and rerun the job.

FABC3737E	ROOT SEGMENT DEFINITION MISMATCH: [SOURCE SEGNAME SOURCE KEY POSITION,LENGTH] "OLD" DMB xxxxxxxx "NEW" DMB xxxxxxxx
------------------	--

Explanation

Program FABCUR1 found that the definition for the root segment in the DMB from NEWACB data set did not match that in the DMB from the OLDACB data set.

System action

FABCUR1 ends with an abend code of 3737.

User response

Verify that the OLDACB and NEWACB DD statements are specified correctly, and the DBDGEN and ACBGEN have been performed correctly for the database being processed. Correct the errors, and rerun the job.

FABC3738E	IMS LEVEL OF MEMBER <i>member_name</i> FROM xxxxxxxxxxxx IS NOT SUPPORTED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR1/FABCUR3/FABCUR5/FABCUR6/FABCUR8 ends with an abend code of 3738.

User response

Make sure that the correct data set is referenced, and that the DBDGEN and ACBGEN, or IMS catalog population have completed successfully for the database being processed. The ACB member may have been assembled with an unsupported release of IMS. Correct the errors, and rerun the job.

FABC3739E	MEMBER <i>member-name</i> FROM NEWACB DATASET IS LOWER IMS RELEASE LEVEL THAN ONE FROM OLDACB DATASET
------------------	--

Explanation

Program FABCUR1 found that IMS release of DMB member from NEWACB data set is lower than the DMB member from OLDACB data set.

System action

FABCUR1 ends with an abend code of 3739.

User response

Make sure that the correct data set is referenced, and that the DBDGEN and ACBGEN have been correctly

performed for the database being processed. Correct the errors, and rerun the job.

FABC3740E	DATABASE STRUCTURE CHANGED; NOT AUTHORIZED
------------------	---

Explanation

Program FABCUR1 determined that the hierarchical structure of the DMB from the NEWACB data set did not match that in the DMB from the OLDACB data set and HIERCHNG=YES/YESFORCE was not specified on the DBDNAME control statement.

System action

FABCUR1 ends with an abend code of 3740.

User response

Verify that the NEWACB DD statement is specified correctly and the required DBDGEN and ACBGEN have been performed correctly. If the structure change is desired, include HIERCHNG=YES/YESFORCE on the DBDNAME control statement. Correct the errors, and rerun the job.

FABC3741E	INVALID DATABASE STRUCTURE CHANGE FOR SEGMENT <i>segname</i>
------------------	---

Explanation

Program FABCUR1 determined that the hierarchical structure of the DMB from the NEWACB data set did not match that in the DMB from the OLDACB data set. The change detected was not one of those allowed by FABCUR1.

System action

FABCUR1 ends with an abend code of 3741.

User response

Verify that the NEWACB DD statement is specified correctly and the DBDGEN and ACBGEN have been performed correctly for the database being processed. For further information about the structure changes allowed by FABCUR1, see the topic "Functions of DEDB Unload" > "Hierarchical structure changes" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide*. Correct the errors, and rerun the job.

FABC3742E	HIERCHNG=YES SPECIFIED W/O AREACTL=ALL
------------------	---

Explanation

An AREACTL control statement with a value of ALL is required if database hierarchy changes are being performed.

System action

Program FABCUR1 ends with an abend code of 3742.

User response

Verify that the NEWACB DD statement is specified correctly and the required DBDGEN and ACBGEN have been performed correctly. If the structure change is desired, specify a value for the AREACTL control statement of ALL. Correct the errors, and rerun the job.

FABC3743E	"MAX UOW BUFFER SPACE" SPECIFICATION IS TOO SMALL. MINIMUM SIZE MUST BE <i>nnnnK</i>.
------------------	--

Explanation

Program FABCUR1 determined that the BASE/DOVF buffer area size parameter on the user-supplied TASKCTL control statement was not large enough to contain at least the DOVF CIs and one (1) BASE CI for the areas being unloaded. The value specified by *nnnn* or larger value must be specified.

System action

FABCUR1 ends with an abend code of 3743.

User response

Review the control statement specifications. Correct the errors, and rerun the job.

FABC3744E	RMODETYPE=G SPECIFIED W/O AREACTL=ALL
------------------	--

Explanation

RMODETYPE=G is specified on the DBDNAME control statement and the NEWACB DD statement is present, but AREACTL=ALL is not specified.

System action

Program FABCUR1 ends with an abend code of 3744.

User response

Verify that the NEWACB DD statement is correct and the required DBDGEN and ACBGEN have been performed correctly. If the type of randomizer is

general, specify a value of all for the AREACTL control statement.

FABC3745E	DATASET AT DDNAME <i>ddname</i> IS NOT FOR AREA <i>xxxxxxxx</i> - CI-SIZE IN DMB <i>xxxxxxxx</i> : <i>xxxx</i> - CI-SIZE OF DATASET: <i>xxxx</i>
------------------	---

Explanation

Program FABCUR1 opened the VSAM data set associated with the *ddname* specified. A comparison of certain key values extracted from the DMB from the OLDACB data set with the contents of the second CI in the VSAM data set specified that the VSAM data set was not the database described by the DMB.

System action

FABCUR1 ends with an abend code of 3745.

User response

Verify that the DD statement specified is correct and that the OLDACB DD statement specifies the correct data set. Correct the errors, and rerun the job.

FABC3745E	DATASET AT DDNAME <i>ddname</i> IS NOT FOR AREA <i>xxxxxxxx</i> - MISMATCH VALIDATING FIELD : <i>xxxxxxxx</i> - VALUE IN DMB <i>xxxxxxxx</i> : <i>xxxx</i> - VALUE IN "DMAC" CI : <i>xxxx</i>
------------------	--

Explanation

Program FABCUR1 opened the VSAM data set associated with the *ddname* specified. A comparison of certain key values extracted from the DMB from the OLDACB data set with the contents of the second CI in the VSAM data set specified that the VSAM data set was not the database described by the DMB.

System action

FABCUR1 ends with an abend code of 3745.

User response

Verify that the DD statement specified is correct and that the OLDACB DD statement specifies the correct data set. Correct the errors, and rerun the job.

FABC3746E	DETECT ERROR IN EQE LIST OF AREANAME: <i>areaname</i> (AREA NO: <i>zzzzz</i>) - NUMBER OF EQE: <i>nn</i>
------------------	---

Explanation

Program FABCUR1 found one or more error control intervals (CIs) extracted from the Error Queue Element (EQE) List in the second CI of the area data set specified.

System action

FABCUR1 ends with an abend code of 3746.

User response

Run the Full Recovery Utility, and rerun the job.

FABC3746E	AREaname: <i>areaname</i> (AREA NO:zzzzz) CANNOT BE PROCESSED DUE TO X'80' SET IN EQE LIST
------------------	---

Explanation

Program FABCUR1 found one or more error control intervals (CIs) extracted from the Error Queue Element (EQE) List in the second CI of the area data set specified.

System action

FABCUR1 ends with an abend code of 3746.

User response

Run the Full Recovery Utility, and rerun the job.

FABC3747E	EEQE DETECTED FOR AREA zzzzz (AREaname: <i>areaname</i>)
------------------	---

Explanation

Program FABCUR1 found that the specified area has an EEQE.

System action

FABCUR1 ends with an abend code of 3747.

User response

Run the Full Recovery Utility, and rerun the job.

FABC3748E	AREA zzzzz (AREaname: <i>areaname</i>) IS RECOVERY NEEDED IN DBRC
------------------	--

Explanation

Program FABCUR1 found that the specified area was recovery needed in DBRC.

System action

FABCUR1 terminates with an abend code of 3748.

User response

Make sure that correct area number was specified. If area number is correct, then run the Full Recovery Utility, and rerun the job.

FABC3749E	NO VALID DATA SET FOUND FOR AREA zzzzz (AREaname: <i>areaname</i>)
------------------	---

Explanation

Program FABCUR1 could not find a valid data set to receive the unloaded specified area.

System action

FABCUR1 terminates with an abend code of 3749.

User response

Get a LIST.RECON output report, identify unused area data set names, and rerun the job.

FABC3750E	"OPEN" FAILED FOR DDNAME <i>ddname subtext</i>
------------------	---

Explanation

OPEN processing failed for the file associated with the indicated DD statement. One of the following subtexts is issued:

- - DD STATEMENT NOT FOUND OR DUMMY/NULLFILE SPECIFIED
- - RECORD LENGTH (LRECL) TOO SMALL (zzzz9 REQUIRED) (zzzz9 SPECIFIED)
- - BLOCK SIZE (BLKSIZE) TOO SMALL (zzzz9 REQUIRED) (zzzz9 SPECIFIED)
- - FAILED BY OS

System action

Program FABCUR3 or FABCUR7 ends with an abend code of 3750.

User response

Based on the reason shown in the message, see the DD statement description for FABCUR3 or FABCUR7. Correct any errors, and rerun the job.

FABC3755E	"OPEN" FAILED - VSAM D/S: DDNAME: <i>ddname</i> (AREA zzzzz9) - DD STATEMENT NOT FOUND
------------------	---

Explanation

Program FABCUR3 found that there were no DD statements for the data set for the specified *ddname*. zzzz9 is the area number.

System action

FABCUR3 ends with an abend code of 3756.

User response

Provide a DD statement that identifies the correct data set for the area to be reloaded, and rerun the job.

FABC3756E	"OPEN" FAILED - VSAM D/S: DDNAME: <i>ddname</i> (AREA: zzzz9) - VSAM ERROR DATA: RETURN CODE: <i>aaa</i> (<i>bb</i>) ACB ERROR: <i>ccc</i> (<i>dd</i>)
------------------	---

Explanation

Program FABCUR3 received a nonzero return code from VSAM when attempting to OPEN the data set for the *ddname* specified. The return code and ACB error code values are shown in both decimal (*aaa*, *ccc*) and hexadecimal (*bb*, *dd*) format. zzzz9 is the area number.

System action

FABCUR3 ends with an abend code of 3756. If duplicate data set is specified on more than two DD statements, this message might be issued.

User response

Make sure that a DD statement is present and that it properly identifies the correct data set for the area to be analyzed. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

FABC3757E	INSUFFICIENT SPACE DEFINED FOR "WORKFILE" DDNAME: DURIWRK
------------------	--

Explanation

During IOVF work data set initialization, program FABCUR3 checked the HALCRBA and determined that insufficient space had been defined for the work data set; that is, an insufficient number of records had been defined.

System action

FABCUR3 abends with a user code of 3757.

User response

Calculate the number of records (that is, CIs) required for the work data set (largest UOW 1 times largest ROOT 2 value). Delete and redefine the work data set with sufficient space, and rerun the reload job.

FABC3758E	"STARTAREA=" VALUE GREATER THAN NO. AREAS DEFINED
------------------	--

Explanation

The STARTAREA= value provided on the control statement is greater than the number of areas defined in the DMB.

System action

Program FABCUR3 abends with a user code of 3758.

User response

Correct the value specified on the control statement, and rerun the reload job.

FABC3759E	"DEVTYPE" FAILED FOR DDNAME <i>ddname</i> (RC=<i>xx</i>)
------------------	---

Explanation

Program FABCUR3 issued an SVC 24 (DEVTYPE) to obtain information about the input/output device associated with *ddname*. The return code specified that the attempt was unsuccessful.

System action

FABCUR3 ends with an abend code of 3759.

User response

Make sure that a DD statement is present for *ddname*, and that it properly identifies the correct data set. Correct any errors, and rerun the job. If this situation persists, report it to systems operations personnel.

FABC3760E	INSUFF. STORAGE FOR: <i>aaaa</i> - INCREASE REGION SIZE
------------------	--

Explanation

Program FABCUR3 issued a GETMAIN macro to allocate storage for the purpose of *aaaa*. The attempt was unsuccessful.

System action

FABCUR3 ends with an abend code of 3760.

User response

Check the reload region size. Increase the REGION parameter in the EXEC statement for FABCUR3 as required, and rerun the job.

FABC3761E	CRITICAL CONTROL CARD ERROR ENCOUNTERED
------------------	--

Explanation

During parsing of the control statement, program FABCUR3 encountered a critical error. The critical error is described by another message.

System action

FABCUR3 ends with an abend code of 3761.

User response

Correct the control statement error, and rerun the job.

FABC3762E	"STARTAREA=" VALUE NOT FOUND IN INPUT FILE
------------------	---

Explanation

Data for the STARTAREA= value provided on the control statement was not found in the input file, or data was read for an area whose number is greater than that specified for the parameter.

System action

Program FABCUR3 abends with a user code of 3762.

User response

Correct the value specified on the control statement, and rerun the reload job. If input to FABCUR3 is a series of concatenated data sets, make sure that they are concatenated in ascending area number order.

FABC3763E	FILE DEFINED BY DDN "DURDBDFN" IS EMPTY
------------------	--

Explanation

Program FABCUR3/FABCUR5/FABCUR7 found that the file which should contain a formatted copy of the DMB (created by FABCUR1/FABCUR5/FABCUR6) is empty.

System action

FABCUR3/FABCUR5/FABCUR7 abends with a user code of 3763.

User response

The file must be re-created by FABCUR5. Be very careful and make sure that the correct dbd names are used for FABCUR5. Rerun the reload job, using the created DURDBDFN file.

FABC3764E	DATA SET FOR AREA: zzzz9 DDNAME: ddname NOT EMPTY
------------------	--

Explanation

Program FABCUR3 found that the VSAM data set for the specified area was not empty. FABCUR3 examined the ENDRBA and found it to be greater than zero. DBFUMINO would issue message DFS2526I under the same conditions. zzzz9 is the area number.

System action

FABCUR3 abends with a user code of 3764. When the same data sets are specified on more than two different area data set control statement (areaxxx), this message might be issued. And if one data set is specified on the different DD statement as reloaded data sets, this message is also issued.

User response

Delete and redefine the VSAM cluster for the specified area. Do not run the DEDB Initialization utility (DBFUMINO). Rerun the reload job.

FABC3765E	DATA SET/DMB CI-SIZE CONFLICT FOR AREA: zzzz9 DDNAME: ddname
------------------	---

Explanation

Program FABCUR3 found that the CI size for the VSAM data set for the specified area did not match the CI size specified in the DBD. DBFUMINO would issue message DFS2509I under the same conditions. zzzz9 is the area number.

System action

FABCUR3 abends with a user code of 3765. If same DD name is specified on more than two different area data statements (areaxxx) as reloaded areas, this message might be issued.

User response

Delete and redefine the VSAM cluster for the specified area with the correct CI size. Rerun the reload job.

**FABC3766E INSUFFICIENT SPACE DEFINED
FOR AREA: zzzz9 DDNAME:
ddname**

Explanation

Program FABCUR3 found (by checking the HALCRBA) that insufficient space has been defined for the specified area. DBFUMINO would issue message DFS2510I under the same conditions. zzzz9 is the area number.

System action

FABCUR3 abends with a user code of 3766.

User response

Delete and redefine the VSAM cluster for the specified area with more space. Rerun the reload job.

**FABC3767E FILE DEFINED BY DDN "DURDATA"
IS EMPTY**

Explanation

Program FABCUR3 found that the file which is supposed to contain the segment data for the area is empty.

System action

FABCUR3 abends with a user code of 3767.

User response

Make sure that the DD statement properly identifies the correct data set for the area to be reloaded. Correct any errors, and rerun the reload job. Empty areas cannot be initialized by FABCUR3.

**FABC3768E DATASET/DMB CI-SIZE CONFLICT
FOR WORKFILE DDNAME:
DURIWRK**

Explanation

Program FABCUR3 found that the CI size of the work data set data set is smaller than the largest CI size specified in the DBD.

System action

FABCUR3 abends with a user code of 3768.

User response

Delete and redefine the work data set. The CI size of the work data set must be equal to the largest area CI size specified in the DBD. Rerun the reload job.

**FABC3769E INPUT DATA SEQUENCE ERROR
(REC# zzz,zzz,zz9)**

Explanation

Program FABCUR3/FABCUR7 found a record sequence error in the input segment data associated with ddname DURDATA.

System action

FABCUR3/FABCUR7 abends with a user code of 3769.

User response

Make sure that the sort for the data set is performed successfully. If input to FABCUR3/FABCUR7 is a series of concatenated data sets, make sure that they are concatenated in an ascending area number order. Correct the error, and rerun the reload job.

FABC3770E TRKCALC FAILED (RC = NN)

Explanation

Program FABCUR3 issued a "TRKCALC" macro to determine if the control interval specified for the area data set or for the DURIWRK data set fits on the device to which the data set is allocated. The control interval size is too large for the device.

System action

FABCUR3 abends with a user code of 3770.

User response

Select a smaller control interval size or a DASD device with a longer track length. Then, rerun the reload job.

**FABC3771E VSAM I/O ERROR - "zzz" AREA
zzzz9 (DDNAME: ddname)
- REQUESTED RBA: eeeeeeee
- VSAM ERROR DATA: RETURN
CODE: aaa (bb)
RPL FDBK: ccc (dd)**

Explanation

Program FABCUR3 received a nonzero return code from VSAM when attempting to get access to either the work data set or area data set. The RBA of the CI being read is shown in hexadecimal format. The return

code and RPL FDBK code values are shown in both decimal (*aaa, ccc*) and hexadecimal (*bb, dd*) format. *zzzz9* is the area number.

System action

FABCUR3 ends with an abend code of 3771.

User response

See *DFSMS Macro Instructions for Data Sets* which describe VSAM administration macros. If this situation persists, contact IBM Software Support.

FABC3772E	DURDATA/DMB DEFINITION MISMATCH - DURDATA AREA#: zzzz9 MAX AREAS IN DMB: zzzz9
------------------	---

Explanation

The area number on the input segment data records is greater than the number of areas defined in the DMB.

System action

Program FABCUR3 abends with a user code of 3772.

User response

This is a serious error. Make sure that the data set associated with ddname DURDBDFN is correct. If it was incorrect, the areas that had already been reloaded (if any) should be deleted and reloaded. If in doubt, correct all JCL and rerun both the unload and reload jobs.

FABC3773E	ERROR "ATTACH"-ING WRITER SUB-TASK (RC = xx)
------------------	---

Explanation

Program FABCUR3 issued an ATTACH macro to attach the writer subtask. A nonzero return code specifies that the attempt was unsuccessful.

System action

FABCUR3 abends with a user code of 3773.

User response

Make sure that the module FABCUR3W is in the load library associated with STEPLIB. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

FABC3774E	ERROR "DETACH"-ING WRITER SUB-TASK (RC = xx)
------------------	---

Explanation

Program FABCUR3 issued a DETACH macro to detach the writer subtask. A nonzero return code specifies that the attempt was unsuccessful.

System action

FABCUR3 abends with a user code of 3774.

User response

Contact IBM Software Support.

FABC3775E	ERROR - MAXIMUM VSAM "RPL'S" EXCEEDED
------------------	--

Explanation

The value specified for IOVFBUF= exceeds 255, the maximum number of RPLs allowed.

System action

Program FABCUR3 abends with a user code of 3775.

User response

Reduce the value specified for the IOVFBUF= parameter, and rerun the reload job.

FABC3776E	ERROR - VSAM INCOMPATIBILITY PROBLEM
------------------	---

Explanation

Program FABCUR3 determined that the length of an RPL generated by a GENCB did not match the length of the DSECT.

System action

FABCUR3 abends with a user code of 3776.

User response

Contact IBM Software Support.

FABC3777E	SDEP FIRST CI RBA DISCREPANCY ERROR FOR AREA zzzzz (AREANAME: areaname)
------------------	--

Explanation

Program FABCUR3 found that the first SDEP CI RBA of the SDEP part of the target area was not the same as the unloaded area with SDEP=PHYSICAL keyword parameter.

System action

FABCUR3 terminates with an abend code of 3777.

User response

VSAM ESDS definition of the target area data set is not correct. Define the target area data set with the same CI size and space definition of the unloaded area data set, and rerun the job.

FABC3778E	AREA zzzzz IS FULL -PROCESSING TERMINATED
------------------	--

Explanation

Program FABCUR3 attempted to allocate an IOVF CI from an overflow unit, but found that all CIs within the overflow unit were in use. FABCUR3 then sequentially searched all overflow units in an attempt to locate an overflow unit that contains available IOVF CIs. All IOVF CIs in all overflow units were allocated. zzzzz is the area number.

System action

FABCUR3 abends with a user code of 3778.

User response

Increase the size of the area, and restart the reload job. Carefully examine the UOW and ROOT parameters.

FABC3779E	SEGCTL TABLE IS FULL - INCREASE NUMBER OF ENTRIES
------------------	--

Explanation

A segment control table is used to retain parentage information when the Insert Limit Count (ILC) feature of program FABCUR1 is used. There is one entry in the table for each ILC case encountered during the processing of a RAP. The default value is 500.

System action

FABCUR3 abends with a user code of 3779.

User response

Increase the number of entries by specifying the TBENTRY= parameter on the control statement. Rerun/restart the reload job.

FABC3780E	VSAM "xxxxxx" ERROR - REG 15: yy REG 0: zz
------------------	---

Explanation

An error was encountered when performing one of the following VSAM functions: GENCB, MODCB, or SHOWCB. The values returned in registers 15 and 0 are shown (in hexadecimal format).

System action

The program abends with a user code of 3780.

User response

"VERIFY" the data set, and rerun/restart the reload job. If this situation persists, contact IBM software Support.

FABC3781E	SEGMENT IN AREA zzzzz [AT RBA xxxxxxx] (SEGCODE yyy) - RETURNED FROM [COMPRESSION ROUTINE cmprname USER EXIT ROUTINE exitname] - EXCEEDS MAX DEFINED LENGTH
------------------	--

Explanation

Program FABCUR1 unload subtask/FABCUR6 determined that the data in the specified segment was, when compressed/expanded by the specified segment edit/compression routine or by the specified user exit routine, longer than the length defined in the DBD member of the OLDACB or NEWACB ACB library. zzzzz is the area number.

System action

FABCUR1 unload subtask/FABCUR6 abends with a user code of 3781.

User response

Make sure that the segment edit/compression routine or the user exit routine is correct. Correct any errors, and rerun the job.

FABC3782E	SEGMENT IN AREA zzzzz AT RBA xxxxxxxx (SEGCODE yyy) RETURNED FROM COMPRESSION ROUTINE cmprname TOO SHORT
------------------	---

Explanation

An unload subtask determined that the data in the specified segment was, when compressed/expanded by the specified segment edit/compression routine, shorter than the length defined in the DBD member of the OLDACB or NEWACB ACB library. zzzzz is the area number.

System action

The unload subtask ends with an abend code of 3782.

User response

Make sure that the segment edit/compression routine is correct. Correct any errors, and rerun the job.

FABC3783E **SEGMENT IN AREA zzzzz [AT RBA
xxxxxxxx] (SEGCODE yyy)
- RETURNED FROM
[COMPRESSION ROUTINE
cmprname | USER EXIT ROUTINE
exitname]
- KEY FIELD MODIFIED**

Explanation

FABCUR1 unload subtask/FABCUR6 determined that the key field in the segment was modified by the indicted segment edit/compression routine or by the specified user exit routine. zzzzz is the area number.

System action

FABCUR1 unload subtask/FABCUR6 abends with a user code of 3783.

User response

Make sure that the segment edit/compression routine or the user exit routine is correct. Correct any errors, and rerun the job.

FABC3784E **INPUT DATA HAS INVALID
COMPRESSION FLAG (REC#
zzz,zzz,zz9)**

Explanation

Program FABCUR3 found one of the listed inconsistencies below among processing flag #1 (USRPFLG1, offset X'47') in the specified unloaded record, the database description table (DDT) flag byte for 'global information' (DDTFLG1, offset X'36'), and the segment description table (SDT) attribute flag byte (SDTFLG1, offset X'16') of the database definition record data set specified by the DURDBDFN DD statement.

- The SDTFCMP flag (X'01') on the SDTFLG1 flag byte in the SDT is off, but the USRPCOMP flag (X'02') on the USRPFLG1 flag byte in the unloaded record is on. That is, no segment edit/compression routine was defined for the segment but the data in the unloaded record was compressed.
- The SDTFCMP flag (X'01') on the SDTFLG1 flag byte in the SDT is on, and the DDTFCMY flag

(X'02') on the DDTFLG1 flag byte in the DDT is off, but the USRPCOMP flag (X'02') on the USRPFLG1 flag byte in the unloaded record is on. That is, segment edit/compression routine was defined for the segment, and the compression request was specified when the area was processed by the DEDB Unload/Reload program, but the unloaded record was not compressed.

- The SDTFCMP flag (X'01') on the SDTFLG1 flag byte in the SDT is on, and the DDTFCMY flag (X'02') on the DDTFLG1 flag byte in the DDT is on, but the USRPCOMP flag (X'02') on the USRPFLG1 flag byte in the unloaded record is off. That is, segment edit/compression routine was defined for the segment, and the compression request was not specified when the area was processed by the DEDB Unload/Reload program, but the unloaded record was compressed.

System action

FABCUR3 ends with an abend code of 3784.

User response

Verify that the unloaded data set specified by the DURDATA DD statement and the database definition record data set specified by the DURDBDFN DD statement are correct. Correct any errors, and rerun the job.

FABC3785E **SEGMENT AT REC#: xxx,xxx,xx9
(AREA zzzzz SEGCODE xxx)
- RETURNED FROM [USER
EXIT ROUTINE exitname
| COMPRESSION ROUTINE
cmprname]
- EXCEEDS MAX DEFINED LENGTH**

Explanation

Program FABCUR3/FABCUR7 determined that the data in the specified segment is, when compressed by the segment edit/compression routine or by the user exit routine, longer than that allowed by the definition for the segment in the database definition record specified by the DURDBDFN DD statement. zzzzz is the area number.

System action

FABCUR3/FABCUR7 abends with a user code of 3785.

User response

Check that the segment edit/compression routine or the user exit routine is correct for the specified segment and that the DURDBDFN DD statement

properly identifies the correct data set for the area to be reloaded. Correct any errors, and rerun the job.

FABC3786E **SEGMENT AT REC#: zzz,zzz,zz9**
(AREA zzz SEGCODE xxx) -
RETURNED FROM COMPRESSION
ROUTINE *comprname* TOO SHORT

Explanation

Program FABCUR3 determined that the data in the specified segment was, when compressed by the specified segment edit/compression routine, shorter than allowed for the database definition record definition specified by the DURDBDFN DD statement.

System action

The unloaded subtask ends with an abend code of 3786.

User response

Check that the segment edit/compression routine is correct for the specified segment and that the DURDBDFN DD statement properly identifies the correct data set for the area to be reloaded. Correct any errors, and rerun the job.

FABC3787E **SEGMENT AT REC#: zzz,zzz,zz9**
(AREA zzzzz SEGCODE xxx) -
RETURNED FROM [COMPRESSION
ROUTINE *comprname* | USER EXIT
ROUTINE *exitname*]
- KEY FIELD MODIFIED

Explanation

Program FABCUR3/FABCUR7 determined that the key field in the specified segment was modified by the specified segment edit/compression routine or by the specified user exit routine. zzzzz is the area number.

System action

FABCUR3/FABCUR7 abends with a user code of 3787.

User response

Check that the segment edit/compression routine or the user exit routine is correct for the specified segment. Correct any errors, and rerun the job.

FABC3788E **SDEP=PHYSICAL SPECIFIED BUT**
- CI SIZE UNMATCH BETWEEN
xxxxxxx AND yyyyyyy
- RANDOMIZER RMODTYPE=G
SPECIFIED/ASSUMED

- AREA NAME SPECIFIED IN
xxxxxxx NOT FOUND IN yyyyyyy

Explanation

Program FABCUR1 found that the SDEP=PHYSICAL keyword parameter is specified, but there is one of the following errors:

- The CI size of an area does not match between OLDACB and NEWACB, or between the current active ACB and the pending ACB in the IMS catalog.
- The REORG keyword is not specified or NEWACB DD is provided with or without the HIERCHNG= keyword parameter.
- The area name defined in the OLDACB was not found in the NEWACB. If IMS-managed ACBs are used, the area name defined in the current active ACB was not found in the pending ACB.

System action

FABCUR1 terminates with an abend code of 3788.

User response

Correct the error, and rerun the job.

FABC3789E **SDEP= CONTROL CARD**
SPECIFIED
- SDEP SEGMENT NOT DEFINED IN
DMB xxxxxxxx FROM yyyyyyy

Explanation

Program FABCUR1 found that the SDEP=LOGICAL|PHYSICAL keyword parameter is specified, but the database specified by the DMB name does not define a SDEP segment.

System action

FABCUR1 terminates with an abend code of 3789.

User response

Make sure that the correct DBDNAME or ACB library, or the IMS catalog is specified. Correct the error, and rerun the job.

FABC3790E **ERROR IN CALL TO RANDOMIZER**
xxxxxxx
- AREA: nnnnn (AREANAME
***areaname*)**
- RMODTYPE=S BUT RANDOMIZED
TO ANOTHER AREA

Explanation

RMODTYPE=S is specified, but the randomizer randomized a record to another area that is different from the original one.

System action

Program FABCUR1 ends with an abend code of 3790.

User response

Check that the load library that contains the randomizer or the RMODTYPE=S option is correct. Correct the error and rerun the job.

FABC3790E	ERROR IN CALL TO RANDOMIZER xxxxxxxx FUNCTION: <i>function_name</i> - RETURN CODE: xxx [REASON CODE: X'xxxxxxxx'] - AREA: zzzzz (AREANAME <i>areaname</i>) - [SEG RBA: xxxxxxxx USR REC#: xxxxxxxx]
------------------	--

Explanation

An unload subtask or a reload job called the specified randomizer module to calculate the new area and RAP number values for a root segment. The values returned were invalid. "FUNCTION" and "REASON CODE" are issued only when FABCUR1 or an unload subtask called the XCI randomizer routine.

System action

The unload subtask or a reload job ends with an abend code of 3790.

User response

Verify that the RMODLIB DD statement specifies the correct data set. Correct the errors, and rerun the job. If this situation persists, report it to database administration personnel.

FABC3791E	INVALID xxx POINTER IN AREA zzzzz (AREANAME <i>areaname</i>) - SOURCE SEG SEG-CD: xxx RBA: xxxxxxxx - TARGET SEG SEG-CD: xxx RBA: xxxxxxxx
------------------	---

Explanation

An IMS pointer in the specified area is in error. The pointer type (RAP, PCF, or PTF), the segment in which

the error was found, and the value of the pointer are shown in the message.

If PTRERROR=BYPASS is specified, program FABCUR1 issues message FABC3791W instead of FABC3791E.

System action

The unload subtask ends with an abend code of 3791.

User response

Consult database administration personnel about procedures for correcting the 'bad' pointer. If an error message was issued, correct the problem, and rerun the unload job.

FABC3791E	INVALID RAP POINTER IN AREA zzzzz (AREANAME <i>areaname</i>) - RAP AT RBA xxxxxx HAS A VALUE OF xxxxxxxx
------------------	---

Explanation

An IMS pointer in the specified area is in error. The pointer type (RAP, PCF, or PTF), the segment in which the error was found, and the value of the pointer are shown in the message.

If PTRERROR=BYPASS is specified, program FABCUR1 issues message FABC3791W instead of FABC3791E.

System action

The unload subtask ends with an abend code of 3791.

User response

Consult database administration personnel about procedures for correcting the 'bad' pointer. If an error message was issued, correct the problem, and rerun the unload job.

FABC3791W	INVALID xxx POINTER IN AREA zzzzz (AREANAME <i>areaname</i>) - SOURCE SEG SEG-CD: xxx RBA: xxxxxxxx - TARGET SEG SEG-CD: xxx RBA: xxxxxxxx
------------------	---

Explanation

An IMS pointer in the specified area is in error. The pointer type (RAP, PCF, or PTF), the segment in which the error was found, and the value of the pointer are shown in the message.

System action

FABCUR1 sets an end-of-job return code of 4 and continues processing.

User response

Consult database administration personnel about procedures for correcting the 'bad' pointer. If an error message was issued, correct the problem, and rerun the unload job.

FABC3791W	INVALID RAP POINTER IN AREA zzzzz (AREANAME areaname) - RAP AT RBA xxxxxx HAS A VALUE OF xxxxxxxx
------------------	--

Explanation

An IMS pointer in the specified area is in error. The pointer type (RAP, PCF, or PTF), the segment in which the error was found, and the value of the pointer are shown in the message.

System action

FABCUR1 sets an end-of-job return code of 4 and continues processing.

User response

Consult database administration personnel about procedures for correcting the 'bad' pointer. If an error message was issued, correct the problem, and rerun the unload job.

FABC3792E	I/O ERROR ATTEMPTING READ OF AREA xxxxx (AREANAME areaname) -REQUESTED RBA: xxxxxxxx -VSAM ERROR DATA: RETURN CODE: xx : RPL "FDBK": xxx (xx)
------------------	--

Explanation

An unload subtask issued a GET for the data set associated with the ddname specified. The return code from VSAM specified that attempt was unsuccessful. The return code and the value of the FDBK field from the RPL are shown.

System action

The unload subtask ends with an abend code of 3792.

User response

See *DFSMS Macro Instructions for Data Sets* that describe VSAM administration macros. If this situation persists, contact IBM Software Support.

FABC3793E	NO OUTPUT FILE PROVIDED FOR DATA FOR AREA zzzzz
------------------	--

Explanation

Program FABCUR1 unload subtask/FABCUR6 was unable to write a segment data record for the specified area because no output File was specified to receive records for that area in the user-supplied FILECTL control statements. zzzzz is the area number.

System action

FABCUR1 unload subtask/FABCUR6 ends with an abend code of 3793.

User response

Review the FILECTL control statements. Correct any errors, and rerun the job.

FABC3794E	INVALID "BLOCK TYPE ID" DETECTED IN AREA zzzzz (AREANAME areaname) - VALUE EXPECTED: X"xx" (xxxx CI) - VALUE FOUND: X"xx" (xxxx CI)
------------------	--

Explanation

An unload subtask read a CI from the specified area. The value in the IMS field known as DBLKBTID was not the value expected.

System action

The unload subtask ends with an abend code of 3794.

User response

Consult database administration personnel about procedures for fixing the bad data. When the problem has been corrected, rerun the unload job.

FABC3795E	SEGMENT IN AREA zzzzz AT RBA xxxxxxxx (SEGCODE xxx) EXCEEDS MAX DEFINED LENGTH
------------------	---

Explanation

An unload subtask determined that the data in the specified segment was longer than allowed by the

definition for that segment in the "output" DMB. zzzzz is the area number.

System action

The unload subtask ends with an abend code of 3795.

User response

Correct the errors, and rerun the job.

FABC3796E	"SUBSET" POINTER ERROR IN AREA zzzzz (AREANAME: areaname)
------------------	--

Explanation

An IMS pointer in the specified area is in error.

If PTRERROR=BYPASS is specified, program FABCUR1 issues message FABC3796W instead of FABC3796E.

System action

The unload subtask ends with an abend code of 3796.

User response

Consult database administration personnel about procedures for fixing the 'SUBSET' pointer error. If the error message was issued, rerun the unload job.

FABC3796W	"SUBSET" POINTER ERROR IN AREA zzzzz (AREANAME: areaname)
------------------	--

Explanation

An IMS pointer in the specified area is in error.

System action

FABCUR1 sets end-of-job return code of 4, and continues processing.

User response

Consult database administration personnel about procedures for fixing the 'SUBSET' pointer error. If the error message was issued, rerun the unload job.

FABC3797E	I/O ERROR FOR OUTPUT DATA SET DDNAME ddname1
------------------	---

Explanation

Program FABCUR1/FABCUR6/FABCUR7 issued a PUT for the ddname1 specified. The PUT operation failed.

System action

FABCUR1/FABCUR6/FABCUR7 ends with an abend code of 3797.

User response

Correct the errors, and rerun the job.

FABC3798E	AREA INFORMATION RECORD FOR AREA zzzzz (AREANAME: areaname) NOT FOUND
------------------	--

Explanation

Area information record for AREA zzz (DDNAME: ddname) could not be found in unloaded segment data records (DD name is DURDATA).

System action

Program FABCUR3 ends with an abend code of 3798.

User response

Correct the errors, and rerun the job.

FABC3799E	DURDBDFN RECORD IS INCORRECT
------------------	---

Explanation

The record format of the data set specified by the DURDBDFN DD statement is incorrect.

System action

Program FABCUR3, FABCUR5, FABCUR7, or FABCUR9 ends with an abend code of 3799.

User response

Correct the errors, and rerun the job. If the problem persists, save the entire run listing, including the dump, the JCL, and all the FPB reports, and contact IBM Software Support.

FABC3800E	GETMAIN FAILED DURING OPEN DCB/ACB
------------------	---

Explanation

A GETMAIN failed during an attempt to get storage for opening a DCB or an ACB. The return code means that the attempt was unsuccessful.

System action

Program FABCUR1 ends with an abend code of 3800.

User response

Increase the region size parameter on the JOB statement or the EXEC statement, and rerun the job.

FABC3801E	DSPSERV xxxxxx FAILURE OCCURRED: RETURN CODE : yyyy REASON CODE : zzzzzzzz
------------------	---

Explanation

Program FABCUR3 found that DSPSERV macro failed. The function code of the DSPSERV macro is shown in xxxxxx and the return code and reason code are shown in yyyy and zzzzzzzz.

System action

FABCUR3 ends with an abend code of 3801.

User response

For further explanation of the error, see the *MVS Programming: Assembler Services Reference*. Correct any errors, and rerun the job.

FABC3802E	ALESERV xxxxxx FAILURE OCCURRED: RETURN CODE : yyyy
------------------	--

Explanation

Program FABCUR3 found that the ALESERV macro failed. The function code of the DSPSERV macro is shown in xxxxxx and the return code is shown in yyyy.

System action

FABCUR3 ends with an abend code of 3802.

User response

For further explanation of the error, see the *MVS Programming: Assembler Services Reference*. Correct any errors, and rerun the job.

FABC3803E	ERROR OCCURRED WHEN PROCESSING DBRC RECON AREA=areaname FUNC=function RC=nn
------------------	--

Explanation

Program FABCUR3 was unable to successfully complete the DBRC call. The meanings of the functions are:

Function
Meaning

SIGNON
Sign-on call

SIGNOFF
Sign-off call

AUTH
Area authorization call

UNAUTH
Area unauthorization call

INIT
INIT function call

INITO
INITO function call

INIT1
INIT1 function call

EOD
EOD function call

EOJ
EOJ function call

The preceding DBRC message explains the reason code.

System action

FABCUR3 ends with an abend code of 3803.

User response

Check the DBRC message preceding this message and follow the response in that message.

FABC3804E	NO DATA SET REGISTERED IN DBRC RECON ADS LIST FOR AREA zzzzz (AREANAME: areaname)
------------------	--

Explanation

Program FABCUR3 found that there was no area data set registered in DBRC for the specified area.

System action

FABCUR3 ends with an abend code of 3804.

User response

Get a LIST.RECON output report, specify an unused area data set name. Specify the name in the *adsname* DD statement, and rerun the job.

FABC3805E	AREA zzzzz (AREANAME: areaname) DDNAME: ddname - NOT SAME DS NAME BETWEEN DD STATEMENT AND DBRC
------------------	--

Explanation

Program FABCUR3 found that the area data set name specified in the ddname DD statement was not the same as the one registered in DBRC.

System action

FABCUR3 ends with ABEND 3805.

User response

Get a LIST.RECON output report and identify an unused area data set name, then specify the name in the DARVSAM DD statement, and rerun the job.

FABC3806E	NO VALID AREA DATA SET SPECIFIED FOR AREA zzzzzz (AREANAME: areaname)
------------------	--

Explanation

Program FABCUR3 found that there was no valid area data set specified for the specified area.

System action

FABCUR3 ends with an abend code of 3806.

User response

If DBRC=Y is specified, get a LIST.RECON output report and identify an unused area data set name, then specify it to the *adsname* DD statement, and rerun the job. If DBRC=N is specified, check and correct the content of the areaxxx DD control statement and adxxxx DD data set(s), and rerun the job.

FABC3807E	DURDBDFN RECORD IS UNSUPPORTED DBT V2 OLD FORMAT
------------------	---

Explanation

Program FABCUR3, FABCUR5, FABCUR7, or FABCUR9 detected that the DURDBDFN record that was specified was an old level of the IMS DBT 2.x format. This old format record cannot be processed due to the lack of definition of minimum segment length.

System action

FABCUR3, FABCUR5, and FABCUR7 end with an abend code of 3807. FABCUR9 ends with a return code of 8.

User response

Re-create the FPB level of the DURDBDFN record file by using the FABCUR5 program with the correct

ACBLIB member. Rerun the job with the re-created DURDBDFN file.

FABC3808E	SDEP CI FORMAT OLDER THAN IMS 6.1 DETECTED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR3 ends with an abend code of 3808, and continues processing.

User response

Re-unload the area whose CI is IMS 6.1 or higher, and rerun the job.

FABC3810E	"OPEN" FAILED FOR DDNAME areaxxx DD STATEMENT NOT FOUND
------------------	--

Explanation

areaxxx DD statement was not found.

System action

Program FABCUR3 ends with an abend code of 3810.

User response

Specify areaxxx DD as reloaded VSAM data set, or areaxxx DD that has DDNAME control statement as multi-area data sets. Rerun the job.

FABC3811E	USER EXIT FABC1IE0 RETURNED WITH NON-ZERO RC
------------------	---

Explanation

User exit routine FABC1IE0 sets nonzero to register 15 and returns to the caller.

System action

Program FABCUR1 ends with an abend code of 3811.

User response

Check the reason of the return code from FABC1IE0.

FABC3812E	USER EXIT FABC3IE0 RETURNED WITH NON-ZERO RC
------------------	---

Explanation

User exit routine FAB3IE0 sets nonzero to register 15 and returns to the caller.

System action

Program FABCUR3 ends with an abend code of 3812.

User response

Check the reason of the return code from FAB3IE0.

FABC3813E	DBDNAME AND xxxxxxxx ARE REQUIRED WHEN RAPERROR=ABEND IS SPECIFIED
------------------	---

Explanation

Self-explanatory.

System action

Program FABCUR3 ends with an abend code of 3813.

User response

See the topic "DEDB Reload SYSIN DD data set" in the *IMS Fast Path Solution Pack: IMS Fast Path Basic Tools User's Guide* for details of the RAPERROR control statement. Correct the error, and rerun the job.

FABC3814E	RAP DATA MISMATCH BETWEEN THE USR FILE AND THE RESULT OF RANDOMIZER - USR REC# : xxx,xxx,xxx - AREA NO IN USR : xxxxx - AREA NO FROM RANDOMIZER : xxxxx - RAP RBA IN USR : xxxxxxxx - RAP RBA FROM RANDOMIZER : xxxxxxxxx
------------------	--

Explanation

Program FABCUR3 found that the RAP data (area number, RAP RBA) in the prefix of the unloaded segment record does not match the result of the randomizer.

System action

FABCUR3 ends with an abend code of 3814.

User response

Check whether the unloaded segment record file, the DEDB definition, or both that are obtained from the

DURDBDFN DD or ACBLIB DD are correct. Correct the error, and rerun the job.

FABC3815E	INCORRECT RAP RBA WAS DETECTED IN THE USR FILE [- RAP RBA IS NOT CI BOUNDARY - RAP RBA IS NOT RAP CI RBA] - USR REC# : xxx,xxx,xxx - RAP RBA : xxxxxxxx
------------------	---

Explanation

Program FABCUR3 found that the RAP RBA in the prefix of the unloaded segment record is not at a CI boundary or is not an RBA of a RAP CI.

System action

FABCUR3 ends with an abend code of 3815.

User response

Check whether the unloaded segment record file, the DEDB definition, or both that are obtained from the DURDBDFN DD or ACBLIB DD are correct. Correct the error, and rerun the job.

FABC3816E	FIRST USR THAT HAS NEW RAP RBA WAS NOT FOR ROOT SEGMENT - USR REC# : xxx,xxx,xxx - RAP RBA : xxxxxxxx - SEGCODE : xxx
------------------	--

Explanation

Program FABCUR3 found that the first unloaded segment record of a new RAP RBA is not a root segment. FABCUR3 expects that all unloaded segment records are sorted in the database hierarchical order and the first segment record of a new RAP RBA is a root segment.

System action

FABCUR3 ends with an abend code of 3816.

User response

Check whether the unloaded segment record file, the DEDB definition, or both that are obtained from the DURDBDFN DD or ACBLIB DD are correct. Correct the error, and rerun the job.

FABC3817E	UNDEFINED SEGMENT CODE FOUND IN THE USR FILE RECORD FOR AREA NO: xxxxx, AREANAME: xxxxxxxxx - USR REC# : xxx,xxx,xxx
------------------	---

- SEGCODE : xxx

Explanation

Program FABCUR3 found that the segment code in the prefix of the unloaded segment record is not defined in the database.

System action

FABCUR3 ends with an abend code of 3817.

User response

Check whether the unloaded segment record file, the DEDB definition, or both that are obtained from the DURDBDFN DD or ACBLIB DD are correct. Correct the error, and rerun the job.

FABC3818E	ROOT SEGMENT KEY LENGTH MISMATCH BETWEEN THE USR FILE AND DBD FOR AREA NO: xxxxx, AREaname: xxxxxxxx - USR REC# : xxx,xxx,xxx - USR KEY LEN : xxxxxxxxx - DBD KEY LEN : xxxxxxxxx
------------------	---

Explanation

Program FABCUR3 found that the root segment key length in the prefix of the unloaded segment record does not match the database definition.

System action

FABCUR3 ends with an abend code of 3818.

User response

Check whether the unloaded segment record file, the DEDB definition, or both that are obtained from the DURDBDFN DD or ACBLIB DD are correct. Correct the error, and rerun the job.

FABC3819E	SEGMENT LEVEL MISMATCH BETWEEN THE USR FILE AND DBD FOR AREA NO: xxxxx, AREaname: xxxxxxx - USR REC# : xxx,xxx,xxx - SEGCODE : xxx - USR SEG LEVEL : xx - DBD SEG LEVEL : xx
------------------	---

Explanation

Program FABCUR3 found that the segment hierarchical level of the segment code in the prefix of the unloaded segment record does not match the database definition.

System action

FABCUR3 ends with an abend code of 3814.

User response

Check the unloaded segment record file and/or the DEDB definition obtained from the DURDBDFN DD or ACBLIB DD is correct. Correct the error, and rerun the job.

FABC3820E	UNSUCCESSFUL DYNALLOC REQUEST (SVC 99) RETURN CODE : xx REASON CODE : yyyy
------------------	---

Explanation

Program FABCUR3 issued an SVC 99 (DYNALLOC) to search information dynamically. The return code specified that the attempt was unsuccessful. The return code is shown in xx, and reason code is shown in yyyy.

System action

FABCUR3 ends with an abend code of 3820.

User response

For further explanation of the error, see the *MVS Programming: Authorized Assembler Services Reference*. Correct any errors, and rerun the job.

FABC3822E	- NOT ENOUGH SDEP SPACE AVAIL IN AREA zzzzz (AREaname: areaname) DUE TO DBD CHANGE - THE FIRST CI RBA IN THE NEW SDEP PART: X'xxxxxxxx' - REQUIRED SDEP SPACE: X'xxxxxxxx'
------------------	---

Explanation

Program FABCUR1 unload subtask identified that the ESDS data set for the indicated area does not have enough space for the SDEP part (range between LB and LE) even if the ESDS data set is defined with the maximum size.

System action

The unload subtask ends with an abend code of 3822.

User response

Change the database definition so that enough space is available for the SDEP part, or delete the SDEP segments so that the required amount of SDEP space is reduced. Then rerun the unload job.

FABC3823E - THE FORMAT OF THE USR FILE IS OLD. REGENERATE THE USR FILE

Explanation

Program FABCUR3 found that the SDEP flag field (USRSDEP) in the area information record of the input USR file is "PN". Such USR files are no longer supported.

System action

FABCUR3 ends with an abend code of 3823.

User response

Regenerate the input USR file by rerunning the unload job with SDEP=PHYSICAL accompanied by a DBD change. Then rerun the reload job.

FABC3825E THE SIZE OF THE RELOADED AREA DATA SET IS MORE THAN 4G BYTES
- AREA NO: *nn*, AREA NAME: *areaname*
- DDNAME: *ddname*, DSNAME: *dsname*

Explanation

The size of the reloaded area data set exceeds 4 GB.

System action

FABCUR3 ends with an abend code of 3825.

User response

Delete and redefine the area data set. Make sure that the size of the ADS does not exceed 4 GB.

FABC3830E IMAGE COPY LOAD MODULE NOT FOUND

Explanation

There were no load modules of the IBM IMS High Performance Image Copy for z/OS in the JOBLIB/STEPLIB DD library.

System action

Program FABCUR3 ends with an abend code of 3830.

User response

Concatenate the load module library data set of IBM IMS High Performance Image Copy for z/OS, and rerun the job.

FABC3831E INCORRECT LEVEL OF IMAGE COPY EXTENSIONS(ICE) LOAD MODULE DETECTED

Explanation

The load modules of the IBM IMS Image Copy Extensions for z/OS utility does not support the interface for Reload.

System action

Program FABCUR3 ends with an abend code of 3831.

User response

Specify the correct level of load module library of IBM IMS Image Copy Extensions for z/OS for the JOBLIB/STEPLIB DD statement.

FABC3832E LOAD FAILED FOR LOAD MODULE *modulename* (ABEND CODE *Sxxxx* / REASON CODE *yyyyyyyy*)

Explanation

Program FABCUR8 issued an SVC 8 (LOAD) to load the module specified by *modulename* into the core. The return code from OS ('Abend Code') specifies that the attempt was unsuccessful.

System action

FABCUR8 ends with an abend code of 3832.

User response

For further information, see the *MVS Programming: Assembler Services Reference*. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

FABC3833E LOAD FAILED FOR DBD MEMBER *dbdname* (ABEND CODE *Sxxxx* / REASON CODE *yyyyyyyy*)

Explanation

Program FABCUR8 issued an SVC 8 (LOAD) to load the DBD member specified by *dbdname* into the core. The return code from OS (*Sxxxx*) specifies that the attempt was unsuccessful.

System action

FABCUR8 ends with an abend code of 3833.

User response

For further information, see the *MVS Programming: Assembler Services Reference*. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

FABC3834E	DBDNAME CONTROL CARD/ [UR7DBDFN DURDBDFN] MISMATCH CTL CARD DBDNAME xxxxxxxx [UR7DBDFN DURDBDFN] DBDNAME xxxxxxxx
------------------	--

Explanation

Program FABCUR3/ FABCUR7 found that the DBD name specified in the DBDNAME control statement and the one specified in the DURDBDFN/UR7DBDFN DD data set do not match.

System action

FABCUR3/FABCUR7 ends with an abend code of 3890.

User response

Check that the DBD name specified on the DBDNAME control statement and the DBD name of the DURDBDFN data set that is specified on the DURDBDFN/UR7DBDFN DD statement are correct. Correct the error, and rerun the job.

FABC3835E	IMS TOOLS CATALOG INTERFACE CANNOT BE USED - UNSUPPORTED IMS RELEASE
------------------	---

Explanation

The IMSCATHLQ=*bsdshlq* parameter is specified on SYSIN, but FABCUR1 or FABCUR3 could not use the IMS Tools Catalog Interface to read the ACB from the IMS directory because the version of IMS is lower than 14.

System action

FABCUR1 or FABCUR3 ends with an abend code of 3835.

User response

Rerun the job using a supported version of IMS.

FABC3836E	IMS TOOLS CATALOG INTERFACE <i>function</i> FUNCTION (DEFINITION=CURRENT PENDING) FAILED
------------------	---

**- RETURN CODE: *rc*, REASON
CODE: *rsn***

Explanation

The IMS Tools Catalog Interface ended with an error. *function* shows the function code of the IMS Tools Catalog Interface. The return code and reason code from the IMS Tools Catalog Interface are shown in *rc* and *rsn*, respectively.

System action

FABCUR1 or FABCUR3 ends with an abend code of U3836.

User response

If the function is OPEN, check if the correct high-level qualifier of the bootstrap data set is specified in the IMSCATHLQ keyword. Otherwise, contact IBM Software Support.

FABC3890E	DBDNAME CONTROL CARD/ DURDBDFN MISMATCH CTL CARD DBDNAME xxxxxxxx DURDBDFN DBDNAME xxxxxxxx
------------------	--

Explanation

Program FABCUR5 found that there is a discrepancy between the DBD name specified in the DBDNAME control statement and the one specified in the DURDBDFN DD data set.

System action

FABCUR5 abends with a user code of 3890.

User response

Verify the correctness of the DBD name specified on the DBDNAME control statement and the DBD name of the DURDBDFN data set specified on the DURDBDFN DD statement. Correct the error, and rerun the job.

FABC3900E	<i>type</i> SEGMENT KEY SEQUENCE ERROR IN AREA <i>nnnnn</i> (AREaname <i>areaname</i>) - SEG - CD: <i>xxx</i> RBA: <i>xxxxxxxxx</i>
------------------	--

Explanation

A segment key sequence in the specified segment of the specified area contains an error. The segment type, ROOT or DDEP, and the segment in which the error was found are shown in the message.

System action

Program FABCUR1 ends with an abend code of 3900.

User response

Consult database administration personnel about procedures for correcting the "bad" sequence field. Correct the problem, and rerun the unload job.

FABC3900E	type SEGMENT KEY SEQUENCE ERROR IN AREA <i>nnnnn</i> (AREANAME <i>areaname</i>) - SEG - CD: <i>xxx</i> - REC#: <i>zzz,zzz,zz9</i>
------------------	--

Explanation

A segment key sequence in the specified segment of the specified area contains an error. The segment type, ROOT or DDEP, and the segment in which the error was found are shown in the message.

System action

Program FABCUR3 ends with an abend code of 3900.

User response

Verify the unloaded file in the DURDATA DD statement. Correct the problem, and rerun the reload job.

FABC3900W	type SEGMENT KEY SEQUENCE ERROR IN AREA <i>nnnnn</i> (AREANAME <i>areaname</i>) - SEG - CD: <i>xxx</i> RBA: <i>xxxxxxxxx</i>
------------------	---

Explanation

If PTRERROR=BYPASS is specified, program FABCUR1 issues message FABC3900W instead of FABC3900E.

System action

FABCUR1 sets an end-of-job return code of 4, and continues processing.

User response

None.

FABC3901E	FILE NUMBER ALREADY EXISTED IN AREA OUTPUT TABLE
------------------	---

Explanation

Program FABCUR1/FABCUR6 tried to set a file number into the area output table (CCIAOUT) but it had been already set.

System action

FABCUR1/FABCUR6 ends with an abend code of 3901.

User response

Contact IBM Software Support.

FABC3902E	LOAD FAILED FOR UNLOAD SUBTASK <i>modname</i> (ABEND CODE <i>Sxxxx</i>/REASON CODE <i>yyyyyyyyy</i>)
------------------	---

Explanation

Program FABCUR1 issued an SVC 8 (LOAD) to bring a copy of the unload subtask into the core. The return code received from OS (*Sxxxx*) specifies that the attempt was unsuccessful. (ABEND CODE and REASON CODE are shown in hexadecimal format)

System action

FABCUR1 ends with an abend code of 3902.

User response

For further explanation of the error, see the *MVS Programming: Assembler Services Reference*. Correct any errors, and rerun the job.

FABC3903E	FILE NUMBER IN AREA OUTPUT TABLE NOT FOUND IN FILE CONTROL TABLE
------------------	---

Explanation

Program FABCUR1 found that the file number in the area output table (CCIAOUT) did not exist in the file control table.

System action

FABCUR1 ends with an abend code of 3903.

User response

Contact IBM Software Support.

FABC3905E	AREA IN <i>xxxxxxxxx</i> AREA <i>nnnnn</i> (AREANAME: <i>areaname</i>) NOT DEFINED IN <i>yyyyyyyyy</i>
------------------	---

Explanation

The area specified is defined in OLDACB but not in NEWACB, or in the current active ACB but not in the pending ACB.

System action

Program FABCUR1 ends with an abend code of 3905.

User response

Make sure that the correct NEWACB data set or the pending ACB is specified. Correct the error, and rerun the job.

FABC3907E	SUBTASK CONTROL ECB POSTED WITH UNEXPECTED REASON. GET UNEXPECTED COMPLETE CODE FROM SUBTASK
------------------	---

Explanation

Program FABCUR1 received an unexpected termination code from the unload subtask.

System action

FABCUR1 ends with an abend code of 3907.

User response

Contact IBM Software Support.

FABC3908E	INSUFFICIENT STORAGE: INCREASE REGION SIZE
------------------	---

Explanation

An unload subtask could not open the VSAM data set for the area data set due to the storage shortage. Program FABCUR1 tried to dispatch the process to another unload subtask, but no subtasks were available.

System action

FABCUR1 ends with an abend code of 3908.

User response

Increase the REGION parameter on the EXEC statement and rerun the job.

FABC3909E	UNLOCK FAILED FOR MESSAGE
------------------	----------------------------------

Explanation

Program FABCUR1 tried to unlock for message resources between subtasks, though they had not been locked.

System action

FABCUR1 ends with an abend code of 3909.

User response

Contact IBM Software Support.

FABC3910E	UNEXPECTED STATUS CODE RETURNED FROM IMS
------------------	---

Explanation

An unexpected IMS status code was returned during a DL/I call.

System action

This message is accompanied by message FABC3912E.

User response

Consult the description of the accompanying message

Module

FABCUR9A

FABC3911E	CALL: <call> STATUS CODE: <sc>
------------------	---

Explanation

An unexpected IMS status code was returned during a DL/I call.

System action

Processing ends with an abend code of 3911.

User response

Provide appropriate action in response to the status as described in *IMS Messages and codes Volume 1*, and resubmit the job.

Module

FABCUR9A

FABC3912E	SEGMENT: <segname> CALL: <call> STATUS CODE: <sc> KEY: <key> REC#: <record number>
------------------	---

Explanation

An unexpected IMS status code was returned during a DL/I call. REC# specifies the relative record within the UNLDREC file.

System action

Processing ends with an abend code of 3912.

User response

Provide appropriate action in response to the status as described in *IMS Messages and codes Volume 1*, and resubmit the job.

Module

FABCUR9A

FABC3914E - EXITRTN=FABCRPCX MUST BE SPECIFIED IN REORG MODE

Explanation

The exit routine FABCRPCX was specified in the Change mode.

System action

Program FABCUR1 abends with user code of 3914.

User response

Remove the NEWACB DD statement, and rerun the job.

FABC3940E DIVISION FAILED - RPL POOL SIZE IS zzzzzzzzzz: NUMBER OF IOVF BUFFERS IS 9z

Explanation

Program FABCUR3 found that the size of the RPL pool for the work data set obtained by GENCB macro was not divisible by the number of IOVF buffers. (The size of the RPL pool and the number of IOVF buffers are shown in decimal format.)

System action

FABCUR3 ends with an abend code of 3940.

User response

Contact IBM Software Support.

FABC3941E RDJFCB FAILED FOR DDNAME ddname (RC=xx)

Explanation

Program FABCUR1/FABCUR3 issued an RDJFCB macro for the ddname specified. The macro failed. (Return code is shown in decimal format)

System action

FABCUR1/FABCUR3 ends with an abend code of 3941.

User response

To determine the cause of the problem specified by the RDJFCB ERROR reason code yyyyyyyy, see *DFSMS Macro Instructions for Data Sets* that explains the error and reason codes of the RDJFCB macro. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

FABC3942E ROOT KEY NOT FOUND IN UTBL

Explanation

Program FABCUR3 found that a root key was not found in the UTBLs for disposing of a segment that had ICL FLAG (USRLCFG) on (X'FF'). The cause might be that there were no segments that had USERPFLG1 on (X'FF') in an unloaded file, and nothing was saved in UTBL.

System action

FABCUR3 ends with an abend code of 3942.

User response

If you create or modify the unloaded file, verify that the unloaded data set is correct. Then, rerun the job. If this situation persists, contact IBM Software Support.

FABC3943E SEGMENT CODE xxx NOT FOUND IN UTBL

Explanation

Program FABCUR3 found different segment codes between the segment unloaded file that had USRPFLG1 on and the segment in UTBLs that had USRLCFLG on (X'FF'). (Segment code in UTBLs is shown in decimal format.)

System action

FABCUR3 ends with an abend code of 3943.

User response

If you created or modified the unloaded file, verify that the unloaded data set is correct. Then, rerun the job. If this situation persists, contact IBM Software Support.

FABC3944E	INSERT FAILED FOR DISPOSING SEGMENT AT THE BASE SECTION - INSERT POSITION NOT ANY OF BASE, DOVF OR IOVF
------------------	--

Explanation

Program FABCUR3 found that the insert position of the disposing segment at the base section was not BASE, DOVF or IOVF.

System action

FABCUR3 ends with an abend code of 3944.

User response

Contact IBM Software Support.

FABC3945E	PARENT SEGMENT CODE FIELD IN USR PREFIX IS INCORRECT FOR AREA NO: <i>nnnnn</i>, AREANAME: <i>areaname</i>. - USR REC# : <i>xxx,xxx,xxx</i> - SEGCODE : <i>xxx</i> - USR PARENT SEGCODE : <i>xxx</i> - DBD PARENT SEGCODE : <i>xxx</i>
------------------	--

Explanation

Program FABCUR3 detected that the segment code in the USRPSCD field of the unloaded segment record specified by the record number (REC#) was not correct. First child segment record must have its parent segment code in the USRPSCD field. Second or subsequent twin segment record must have its same segment code.

System action

FABCUR3 ends with an abend code of 3945.

User response

Check the content of the unloaded segment record specified by the record number (REC#), correct the value of the USRPSCD field, and rerun the job.

FABC3946E	INSERT FAILED FOR DISPOSING SEGMENT AT THE OVERFLOW SECTION - INSERT POSITION NEITHER A DOVF NOR IOVF
------------------	--

Explanation

Program FABCUR3 found that the insert position of the disposing segment at the overflow section was neither a DOVF nor IOVF.

System action

FABCUR3 ends with an abend code of 3946.

User response

Contact IBM Software Support.

FABC3948E	THE PLACE OF THE PARENT SEGMENT NOT ANY OF BASE, DOVF OR IOVF
------------------	--

Explanation

Program FABCUR3 found that the place of the parent segment was not BASE, DOVF, or IOVF, when the insert segment's pointer was set into its parent segment.

System action

FABCUR3 ends with an abend code of 3948.

User response

Verify that the unloaded data set is correct. If you modified or created the unloaded file, the cause might be that the USRPSCD of the first occurrence of dependent segment is incorrect. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

FABC3950E	CALL FUNCTION "<i>function</i>" ERROR (FABCUR6/FABCUR7) - "<i>function</i>" CALL PROCESSED ALREADY - "<i>function</i>" CALL NOT PROCESSED YET - "INI<i>x</i>" CALL INITIATED BUT "GET<i>x</i>" CALL ISSUED - STATUS "GB" RETURNED ALREADY - "INIT" CALL DBNAME PARAMETER NOT PROVIDED - "<i>function</i>" CALL UNKNOWN PARAMETER SPECIFIED - "PUT" CALL NO I/O AREA SPECIFIED - "PUT" CALL NO SEGMENT NAME FOUND - "PUT" CALL SEGMENT DATA LENGTH TOO SHORT
------------------	--

- "PUT" CALL SEGMENT DATA LENGTH TOO LONG
- "PUT" CALL INVALID SEGMENT SUBSET POINTER
- "GETx" CALL NO STATUS CODE AREA SPECIFIED
- "GETx" CALL NO I/O AREA SPECIFIED
- "GETx" CALL NO EX. I/O AREA SPECIFIED
- "EOF" CALL UNKNOWN FUNCTION CODE PARAMETER SPECIFIED
- UNKNOWN FUNCTION CODE

Explanation

Program FABCUR6/FABCUR7 was called from the application but the call failed due to the reason described by the subtext.

System action

FABCUR6/FABCUR7 ends with an abend code of 3950.

User response

Correct the application program logic to call FABCUR6/7 correctly, and rerun the job.

FABC3951W	EXIT ROUTINE <i>exit-name</i> RETURNED STATUS CODE E1 - first 80 bytes characters of the message that user exit routine returned - subsequent 48 bytes characters of the message that user exit routine returned
------------------	---

Explanation

Program FABCUR1 unload subtask/FABCUR3/FBCUR6/FABCUR7 got the status code E1 from the user exit routine *exit-name* specified by the EXITRTN= control statement.

System action

FABCUR1/FABCUR3/FABCUR6/FABCUR7 unload subtask sets an end-of-job return code of 4.

User response

Investigate why the user exit routine returned the status code E1. Correct the problem, and rerun the job.

FABC3951E	EXIT ROUTINE <i>exit-name</i> RETURNED STATUS CODE E2
------------------	--

- first 80 bytes characters of the message that user exit routine returned
- subsequent 48 bytes characters of the message that user exit routine returned

Explanation

Program FABCUR1 unload subtask/FABCUR3/FBCUR6/FABCUR7 got the status code E2 from the user exit routine *exit-name* specified by the EXITRTN= control statement.

System action

FABCUR1/FABCUR3/FABCUR6/FABCUR7 unload subtask ends with an abend code of 3951.

User response

Investigate why the user exit routine returned the status code E2. Correct the problem, and rerun the job.

FABC3952E	EXIT ROUTINE <i>exit-name</i> RETURNED INVALID STATUS CODE cc (X'xxxx')
------------------	--

Explanation

Program FABCUR1/FABCUR3/FBCUR6/FABCUR7 unload subtask got the invalid status code specified by cc (X'xxxx') from the user exit routine *exit-name* specified by the EXITRTN= control statement.

System action

FABCUR1 unload subtask /FABCUR3/FABCUR6/FABCUR7 ends with an abend code of 3952.

User response

Investigate the logic of the user exit routine. Correct the exit routine, and rerun the job.

FABC3953E	ERROR IN CALL TO RANDOMIZER xxxxxxx subtext
------------------	--

Explanation

Program FABCUR6 called the specified randomizer module to calculate the new area and RAP number values for a root segment. The values returned were invalid. One of the following subtexts is issued:

- FUNCTION: *function_name*
 - RETURN CODE: xxxx REASON CODE: X'xxxxxxxx'
- - INVALID AREA #, RAP # VALUES RETURNED FOR SEG AT RBA xxxxxxxx

(VALUES RETURNED: AREA #: xxxxx, RAP #: xxx,xxx,xxx)

"FUNCTION" and "REASON CODE" are issued only when FABCUR6 called the XCI randomizer routine.

System action

FABCUR6 ends with an abend code of 3953.

User response

Verify that the RMODLIB DD statement specifies the correct data set. Correct the errors, and rerun the job. If this situation persists, report it to database administration personnel.

FABC3954E	INPUT DATA ERROR (DDNAME: <i>ddname</i> REC# <i>zzz,zzz,zz9</i>) - SEGMENT CODE <i>zzzzz</i> NOT FOUND - HIERARCHY LEVEL INCORRECT (DURDBDFN: <i>nn</i> UNLOAD FILE RECORD: <i>nn</i>) - PARENT SEGMENT CODE INCORRECT (DURDBDFN: <i>nn</i> UNLOAD FILE RECORD: <i>nn</i>) - ROOT KEY LENGTH INCORRECT (DURDBDFN: <i>nn</i> UNLOAD FILE RECORD: <i>nn</i>) - DATA LENGTH ERROR (DURDBDFN MAX: <i>nnnnn</i> MIN: <i>nnnnn</i> UNLOAD FILE RECORD: <i>nnnnn</i>) - SUBSET POINTER INCORRECT - AREA <i>zzzzz</i> NOT DEFINED IN DMB - AREA INFORMATION RECORD FOR AREA <i>zzzzz</i> NOT FOUND - FIRST DATA NOT CORRECT AREA INFORMATION RECORD
------------------	--

Explanation

Self-explanatory. zzzzz is the area number.

System action

Program FABCUR7 ends with an abend code of 3954.

User response

Make sure that the correct data set is specified. If the data set is correct, make sure that the data set is sorted successfully. If input to FABCUR7 is a series of concatenated data sets, make sure that they are concatenated in an ascending area number order. Correct the error, and rerun the reload job.

FABC3955E	SEGMENT HIERARCHICAL SEQUENCE ERROR DETECTED
------------------	---

Explanation

Program FABCUR6 found that the segment provided by the application program was not in hierarchical sequence.

System action

FABCUR6 ends with an abend code of 3955.

User response

Verify that segment data used by the application program for FABCUR6 input is correct. Correct the errors, and rerun the job.

FABC3956E	EXIT ROUTINE <i>exit-name</i> RETURNED STATUS CODE <i>cc</i>(X'<i>zzzzz</i>') BUT 'T2' EXPECTED
------------------	--

Explanation

Program FABCUR1 unload subtask got the status code specified by *cc* (X'*zzzzz*') from the user exit routine *exit-name* specified on the EXITRTN= control statement. The subtask expected the status code T2 because the exit routine returned the T2 status code for the previous parent/twin segment.

System action

FABCUR1 unload subtask ends with an abend code of 3956.

User response

Investigate the logic of the user exit routine. Correct the exit routine, and rerun the job.

FABC3957E	- UNLOADED SEGMENT RECORD LENGTH EXCEEDS LRECL FOR DDNAME: <i>ddname</i> - USR LL: <i>nnnnn</i> - LRECL: <i>mmmmm</i>
------------------	--

Explanation

The length of the unloaded segment record is greater than LRECL of the output file specified by the *ddname*. A segment might be expanded by compression operation of the edit/compression routine and exceeded the maximum length that is defined in DBD.

System action

Program FABCUR1/FABCUR6 abends with user code of 3957.

User response

Increase the LRECL of the output data set specified by the *ddname* at least 10 bytes, and if specified, do the same for another output data set, and rerun the job.

FABC3958E - AREA INFORMATION RECORD FOUND WHEN AREC=N WAS SPECIFIED.

Explanation

Program FABCUR3 found the area information record in an unloaded file when AREC=N was specified on the EXEC parameter.

System action

FABCUR3 abends with a user abend code of 3958.

User response

Specify a correct unloaded file or specify AREC=Y on the EXEC parameter, and rerun the job.

FABC3959E - AREC=N IS NOT ALLOWED FOR SDEP=PHYSICAL WITH RELOCATION MODE

Explanation

AREC=N was specified with SDEP=PHYSICAL and RMODTYPE=S when both OLDACB DD and NEWACB DD were specified. Because it intends to relocate the SDEP segments at reload time, AREC=N is not allowed. An area information record is essential to relocate SDEP segments at reload.

System action

Program FABCUR1 abends with an abend code of 3959.

User response

Correct the combination of the EXEC parameter, control statements, and ACB DD statements, and rerun the job.

FABC3960E - AREA INFORMATION RECORD FOUND WHEN AREA_INFORMATION_RECORD=N O WAS SPECIFIED

Explanation

Program FABCUR7 found the area information record in an unloaded file when the AREA_INFORMATION_RECORD=NO control statement was specified.

System action

FABCUR7 abends with user code of 3960.

User response

Specify a correct unloaded file or specify the AREA_INFORMATION_RECORD=YES control statement, and rerun the job.

FABC3961E - SPECIAL RECORD FOUND WHEN AREC=Y WAS SPECIFIED

Explanation

Program FABCUR3 found the special record in unloaded segment file when AREC=Y was specified on EXEC parameter.

System action

FABCUR3 abends with user code of 3961.

User response

Specify a correct unloaded file or specify AREC=N on the EXEC parameter, and rerun the job.

FABC3962E - SPECIAL RECORD FOUND WHEN AREA_INFORMATION_RECORD=YES WAS SPECIFIED

Explanation

Program FABCUR7 found the special record in unloaded segment file when AREA_INFORMATION_RECORD=YES was specified.

System action

FABCUR7 abends with user code of 3962.

User response

Specify a correct unloaded file or specify the AREA_INFORMATION_RECORD=NO control statement, and rerun the job.

FABC3989E FIRST INSERTED SEGMENT NOT ROOT SEGMENT
- AREA: zzzzz SEG-CD: xxx RBA: xxxxxxxx

Explanation

Program FABCUR3 found that the first inserted segment of the CI in the base section was not a root segment. If the unload file was not sorted before reloading, this abend could occur.

System action

FABCUR3 ends with an abend code of 3989.

User response

Verify that the UNLOAD FILE is sorted. If it is not, sort it and rerun the job.

FAB4095E RECON ACCESS FAILED. subtext

Explanation

An error was detected in the RECON access processing. One of the following subtexts is issued:

FABD messages

The following information is about messages and codes that begin with FABD.

**FABD0100I OBTAINED DB DEFINITIONS
FROM resource**

Explanation

This message indicates the resource (ACB library or IMS directory) where FABCRMIF or FABDRMIF obtained database definitions from.

System action

FABCRMIF or FABDRMIF continues processing.

User response

None. This message is informational.

**FABD3650E DEVTYPE FAILED FOR DDNAME:
ddname (RC = xx)**

Explanation

Program FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO issued a DEVTYPE macro for the MODSTAT data set ddname specified. The macro failed. (Return code is shown in decimal format)

System action

FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO ends with an abend code of 3650.

- DBRC LIST COMMAND IS NOT COMPLETED.
RC=xxxxxxx
- SYSPRINT DD FOR DBRC LIST COMMAND IS SPECIFIED AS DUMMY
- INTERNAL ERROR OCCURRED
- FUNC=ffffffff RETURN CODE=xxxxxxx REASON
CODE=xxxxxxx
KEYS: DBD=ddname DDN=ddname
KEYTYPE=xxxxxxxxxxx

System action

Program FABCUR1 or FABCUR3 ends with an abend code of 4095.

User response

Check the DBRC message preceding this message. Follow the response in that message, and rerun the job.

User response

To determine the cause of the problem specified by the DEVTYPE return code xx, see *DFSMS DFSMSdfp Advanced Services*, which explains the error return codes of the DEVTYPE macro. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

**FABD3651E OPEN FAILED FOR DDNAME
ddname**

Explanation

Program FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO issued an SVC 19 (OPEN) for the file associated with the MODSTAT data set DD statement specified. The OPEN was not successful.

System action

FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO ends with an abend code of 3651.

User response

Make sure that the DD statement specified is present in the JCL stream being run, and that it properly specifies the correct data set. Correct any errors, and rerun the job.

**FABD3652E I/O ERROR FOR INPUT DATA SET
DDNAME ddname**

Explanation

Program FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO issued a GET for the MODSTAT data set ddname specified. The GET operation failed.

System action

FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO ends with an abend code of 3652.

User response

Correct the errors, and rerun the job.

FABD3653E	NO RECORD FOUND IN DATA SET DDNAME <i>ddname</i>
------------------	---

Explanation

Program FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO found that the file for the MODSTAT data set ddname specified is empty.

System action

FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO ends with an abend code of 3653.

User response

Make sure that the DD statement properly identifies the correct data set for the MODSTAT. Correct any errors, and rerun the job.

FABD3654E	INCORRECT RECORD FOUND IN DATA SET DDNAME <i>ddname</i>
------------------	--

Explanation

Program FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO found that the record for the MODSTAT data set ddname specified is incorrect.

System action

FABADA1/FABCUR5/FABCUR7/FABCRMIF/FABDRMIO ends with an abend code of 3654.

User response

Make sure that the DD statement properly identifies the correct data set for the MODSTAT. Correct any errors, and rerun the job.

FABD3655E	INVALID CONTROL CARD ENCOUNTERED - UNKNOWN KEYWORD
------------------	---

- VALUE FOR THE *keyword* KEYWORD IS INVALID

Explanation

The user-supplied control statement was found to contain one or more errors.

System action

FABCRMIF or FABDRMIF ends with an abend code of 3655.

User response

Correct the control statement and rerun the job.

FABD3656E	IMS TOOLS CATALOG INTERFACE <i>function</i> FUNCTION (DEFINITION=CURRENT PENDING) FAILED - RETURN CODE: <i>rc</i>, REASON CODE: <i>rsn</i>
------------------	--

Explanation

IMS Tools Catalog Interface ended with an error. *function* shows the function code of IMS Tools Catalog Interface. The return code and reason code from IMS Tools Catalog Interface are shown in *rc* and *rsn*, respectively.

System action

The job ends with an abend code of U3656.

User response

If the function is OPEN, check if the correct high-level qualifier of the bootstrap data set is specified for the IMSCATHLQ keyword. Otherwise, contact IBM Software Support.

FABD3661E	INVALID PARM LIST IN CALL TO FABCRMIF/FABDRMIO - UNKNOWN VALUE FOR "FUNCTION" PARAMETER - 'FUNC' = "INIT"; "DBDNAME" PARAMETER NOT FOUND - 'FUNC' = "CALC"; TOO FEW PARM'S SPECIFIED
------------------	---

Explanation

Program FABCRMIF/FABDRMIO determined that the parameter list specified by the calling program was incorrect.

System action

FABCRMIF/FABDRMIO ends with an abend code of 4011.

User response

Correct the CALL specifications in the program being used to invoke FABCRMIF/FABDRMIO. Rerun the job.

FABD3662E **PROCESSING FAILED FOR xxxxxx MEMBER**
- xxxxxx MEMBER IS NOT A DEDB DMB
- MEMBER NAME NOT EQUAL DEFINED DATABASE NAME
- ddname DD STATEMENT NOT FOUND
- MEMBER NOT FOUND IN xxxxxx
- INSUFFICIENT STORAGE
- INVALID PARAMETER LIST IN CALL TO FABCGDD
- IMS LEVEL OF ACB MEMBER xxxxxx NOT SUPPORTED
- ACB MEMBER VRSDSRF IS NOT SAME IMS LEVEL AS DBT LIBRARY
- NO. SEGS DEFINED EXCEEDS ALLOWED MAX.
- RC = XX

Explanation

To obtain DMB information of the database that is being processed, program FABCRMIF/FABDMRIF called either the FABAGDD program to obtain DMB information from the ACBLIB library or the FABAGDD2 program to obtain DMB information from the IMS directory. The return code indicates that the attempt to do so was unsuccessful.

System action

FABCRMIF/FABDMRIF ends with an abend code of 4011.

User response

Ensure that the files associated with the DD statements ACBLIB or the IMS directory are correctly specified. Also, ensure that the ACBGEN and DBDGEN or IMS catalog population were correctly performed for the database being analyzed. Correct any errors, and rerun the job.

If "- INVALID PARAMETER LIST IN CALL TO FABAGDD" is shown, contact IBM Software Support.

FABD3663E **ERROR IN "CALC" CALL TO FABCRMIF/FABDRMIO**
- "INIT"-IALIZATION HAS NOT BEEN PERFORMED

Explanation

Self-explanatory.

System action

Program FABCRMIF/FABDRMIO ends with an abend code of 4011.

User response

Correct the call sequence in the program being used to invoke FABCRMIF/FABDRMIO, and rerun the job.

FABD3664E **macro-name FAILED FOR DMB MEMBER member-name FOR DDNAME ddname (RC = rr : REASON = zz)**

Explanation

Program FABCRMIF/FABDRMIO issued the macro specified (*macro-name*) to access the DMB member (*member-name*) in the data set specified (*ddname*). The z/OS return code and reason code mean that the attempt was unsuccessful.

System action

FABCRMIF/FABDRMIO ends with an abend code of 4011.

User response

For further information, see the *MVS Programming: Assembler Services Reference*. Correct the problem, and rerun the job. If this situation persists, contact IBM Software Support.

FABD3665E **OPEN FAILED FOR DDNAME ddname**

Explanation

Program FABCRMIF/FABDRMIO issued an SVC 19 (OPEN) for the file associated with the DD statement specified. The OPEN was not successful.

System action

FABCRMIF/FABDRMIO ends with an abend code of 4011.

User response

Make sure that the DD statement specified is present in the JCL stream being run, and that it properly specifies the correct data set. Correct any errors, and rerun the job.

FABD3666E	LOAD FAILED FOR RANDOMIZER ROUTINE <i>rmodname</i> (ABEND <i>Sxxx</i> / REASON CODE <i>xxxxxxxx</i>)
------------------	---

Explanation

Program FABCRMIF/FABDRMIO issued an SVC 8 (LOAD) to bring into storage a copy of the randomizer routine specified in the DMB for the database being processed. The return code received from OS means that the attempt failed. The return code (reason code) and abend code returned by OS are shown in the message.

System action

FABCRMIF/FABDRMIO ends with an abend code of 4011.

User response

For further explanation of the error, see the *MVS Programming: Assembler Services Reference*. Correct any errors, and rerun the job.

FABD3667E	DEVTYPE FAILED FOR DDNAME:<i>ddname</i> RC = <i>xx</i>)
------------------	--

Explanation

Program FABCRMIF/FABDRMIO issued a DEVTYPE macro for the *ddname* specified. The macro failed. (Return code is shown in decimal format).

System action

FABCRMIF/FABDRMIO ends with an abend code of 4011.

User response

To determine the cause of the problem specified by the DEVTYPE return code *xx*, see *DFSMS DFSMSdfp Advanced Services*, which explains the error return codes of the DEVTYPE macro. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

FABD3668E	ERROR IN "INIT" CALL TO FABCRMIF
------------------	---

"INIT" -IALIZATION HAS ALREADY BEEN PERFORMED FOR DMB MEMBER *member-name*

Explanation

Self-explanatory.

System action

Program FABCRMIF ends with an abend code of 4011.

User response

Correct the call sequence in the program that invokes FABCRMIF, and rerun the job.

FABD3669E	ERROR IN "INIT" CALL TO FABCRMIF. MORE THAN 16 DMB MEMBERS REQUESTED FOR DMB MEMBER <i>member-name</i>
------------------	---

Explanation

Program FABCRMIF found that 16 DMB members have already been initialized.

System action

FABCRMIF ends with an abend code of 4011.

User response

FABCRMIF supports up to 16 different DMB members. Correct the program invoking FABCRMIF, and rerun the job.

FABD3670E	ERROR IN CALL TO RANDOMIZER ROUTINE <i>rmodname</i> FUNCTION: <i>function_name</i> - RETURN CODE: <i>xxx</i> REASON CODE: X'<i>xxxxxxxx</i>'
------------------	---

Explanation

FABCRMIF/FABDRMIO invoked the XCI randomizer routine with a function code of 'INIT', 'CALC', or 'TERM', and then returned with error. "FUNCTION" and "REASON CODE" are issued only when FABCRMIF/FABDRMIO invoked the XCI randomizer routine. Function is one of 'INITIALIZATION CALL', 'RANDOMIZING CALL', or 'TERMINATION CALL'.

System action

FABCRMIF/FABDRMIO ends with an abend code of 4011.

User response

Make sure that the RMODLIB DD statement properly identifies the correct data set, and that the randomizer routine has been correctly added, assembled, and link-edited. Correct any errors, and rerun the job.

FABD3671E	LOAD FAILED FOR LOAD MODULE FABCRMIX/FABCRMIZ (ABEND Sxxx/REASON CODE xxxxxxxxx)
------------------	---

Explanation

Program FABCRMIF/FABDRMIO/FABDRMIF issued an SVC 8 (LOAD) to bring into storage a copy of FABCRMIX/FABCRMIZ. The return code received from OS means that the attempt failed. The return code (reason code) and abend code returned by OS are shown in the message.

System action

FABCRMIF/FABDRMIO/FABDRMIF ends with an abend code of 4011.

User response

Make sure that the IMS HP Fast Path Utilities load module library is concatenated to the JOBLIB/STEPLIB DD statement. If it is concatenated, see the *MVS Programming: Assembler Services Reference* for a further explanation of the problem. Correct any errors, and rerun the job.

FABD3672E	INVALID keyword= KEYWORD SITE DEFAULT ERROR DETECTED
------------------	---

Explanation

An incorrect parameter was specified for the FABAO1M, FABCOP1M, FABCOP3M, FABCOP6M, or FABCOP9M macro keyword.

System action

The assemble step ends with a return code of 8.

User response

Correct the error, and rerun the job.

FABD3673E	TABLESET=DSECT/SYSTEM AND ANY OTHER KEYWORDS ARE MUTUALLY EXCLUSIVE
------------------	--

Explanation

TABLESET=DSECT/SYSTEM cannot be specified with any other keyword parameters. TABLESET=DSECT/SYSTEM is for system use only. TABLESET=DSECT/SYSTEM must not be specified to define the site default table.

System action

The assemble step ends with a return code of 8.

User response

Correct the error, and rerun the job.

FABD3674E	NO KEYWORD IS SPECIFIED FOR SITE DEFAULT TABLE
------------------	---

Explanation

No keyword is specified for the FABAO1M, FABCOP1M, FABCOP3M, FABCOP6M, or FABCOP9M macro.

System action

The assemble step ends with a return code of 8.

User response

Correct the error, and rerun the job.

FABD3675I	keyword= PARAMETER IS IGNORED BECAUSE DEFAULT VALUE IS SPECIFIED
------------------	---

Explanation

A keyword= parameter was specified that is the same as the system default value. The FABAO1M, FABCOP1M, FABCOP3M, FABCOP6M, or FABCOP9M macro skips generating an entry of the site default table for keyword=.

System action

The assemble step continues normal processing.

User response

None. This message is informational.

FABD3676E	macro-name MACRO SPECIFIED MORE THAN ONCE
------------------	--

Explanation

The FABAO1M, FABCOP1M, FABCOP3M, FABCOP6M, or FABCOP9M macro was specified more than once.

This macro must be specified only once when TABLESET=USER is specified (default value).

System action

The assemble step ends with a return code of 8.

User response

Correct the error, and rerun the job.

FABD3677E	[IMSCOMP=/DLICOMP= AREA_INFORMATION_RECORD=/ AIR= INPUT=/FORMAT=] KEYWORDS ARE MUTUALLY EXCLUSIVE
------------------	--

Explanation

The specified keywords cannot be used together.

System action

The assemble step ends with a return code of 8.

User response

Correct the error, and rerun the job.

FABD3678E	WHEN LRECL=SEGTFMT IS SPECIFIED, FORMAT=TFMT HAS TO BE SPECIFIED
------------------	---

Explanation

You have to specify FORMAR=TFMT when LRECL=SEGTFMT is specified.

System action

The assemble step ends with a return code of 8.

User response

Correct the error, and rerun the job.

FABD3690E	INVALID MESSAGE NUMBER DETECTED MESSAGE NO. <i>nnnn</i> -ERROR NO.(HEX) IS IN REG15
------------------	--

Explanation

While processing an error message, an invalid message number in register 15 was detected. This is an internal error.

System action

Program FABADA1/FABCUR1 ends with an abend code of 3690.

User response

Contact IBM Software Support.

FABD3693E	OBTAIN FAILED FOR VOL=SER=aaaaaa
------------------	---

Explanation

Program FABADA1 or FABCUR1 issued an OBTAIN macro to reserve the volume *aaaaaa*. The attempt was unsuccessful.

System action

FABADA1 or FABCUR1 ends with an abend code of 3693.

User response

Check the volume, fix the problem, and rerun the job.

FABD3694E	INSUFFICIENT STORAGE FOR <i>aaaa</i> - INCREASE REGION SIZE
------------------	--

Explanation

Program FABADA1 or FABCUR1 issued a GETMAIN macro to allocate storage for the purpose of *aaaa*. The attempt was unsuccessful.

System action

FABADA1 or FABCUR1 ends with an abend code of 3694.

User response

Check the region size, increase the REGION parameter in the EXEC statement for FABADA1 or FABCUR1 as required, and rerun the job.

FABD3700E	TWO USABLE RECON DATA SETS ARE NOT PROVIDED FOR ERRORS RECON1 DD: <i>subtext</i> RECON2 DD: <i>subtext</i> RECON3 DD: <i>subtext</i>
------------------	---

Explanation

For the explanation of the case for each subtext, see the corresponding explanation.

System action

For the system action of the case for each subtext, see the corresponding system action.

User response

For the user response of the case for each subtext, see the corresponding user response.

subtext	DEVTYPE FAILED (RC=xx)
----------------	-------------------------------

Explanation

Program FABADA1 or FABCUR1 issued a DEVTYPE macro for the DD name specified. The macro failed. (Return code is shown in hexadecimal format.)

System action

FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

To determine the cause of the problem, see *DFSMS DFSMSdfp Advanced Services*, which explains the error return codes of the DEVTYPE macro. Correct any errors and rerun the job. If this situation persists, contact IBM Software Support.

subtext	RDJFCB FAILED (RC=xx)
----------------	------------------------------

Explanation

Program FABADA1 or FABCUR1 issued an RDJFCB macro for the DD name specified. The macro failed. (Return code is shown in hexadecimal format.)

System action

FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

To determine the cause of the problem, see *DFSMS DFSMSdfp Advanced Services*, which explains the error return codes of the RDJFCB macro. Correct any errors and rerun the job. If this situation persists, contact IBM Software Support.

subtext	SHOWCB FAILED (REG15=xx REG0=yy)
----------------	---

Explanation

Program FABADA1 or FABCUR1 issued a VSAM SHOWCB macro. The macro failed with return code xx

and reason code yy. (The content of the register 15 and register 0 are shown in hexadecimal format.)

System action

FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

To determine the cause of the problem, see *DFSMS Macro Instructions for Data Sets*, which explains the error return codes of the SHOWCB macro. Correct any errors and rerun the job. If this situation persists, contact IBM Software Support.

subtext	OPEN FAILED (RC=xx(XX) REASON CODE=yyy(YY))
----------------	--

Explanation

Program FABADA1 or FABCUR1 issued a VSAM OPEN macro for the RECON data set specified. The macro failed with return code xx in decimal format (XX in hexadecimal format) and reason code yyy in decimal format (YY in hexadecimal format).

System action

FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

To determine the cause of the problem, see *DFSMS Macro Instructions for Data Sets*, which explains the error return codes of the VSAM OPEN macro. Correct any errors and rerun the job. If this situation persists, contact IBM Software Support.

subtext	MODCB FAILED (REG15=xx REG0=yy)
----------------	--

Explanation

Program FABADA1 or FABCUR1 issued a VSAM MODCB macro. The macro failed with return code xx and reason code yy. (The content of register 15 and register 0 are shown in hexadecimal format.)

System action

FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

To determine the cause of the problem, see *DFSMS Macro Instructions for Data Sets*, which explains the

error return codes of the MODCB macro. Correct any errors and rerun the job. If this situation persists, contact IBM Software Support.

subtext	INCORRECT IMS RELEASE LEVEL RECON DATA SET IS USED
----------------	---

Explanation

The data set used for the DD name was not a correct IMS release level of the RECON data set.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Specify the correct IMS release level of the RECON data set and rerun the job.

subtext	NO RECON HEADER/EXTENSION RECORD FOUND
----------------	---

Explanation

The data set used for the DD name was not a RECON data set or a correct IMS release level of the RECON data set.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Specify the correct IMS release level of the RECON data set and rerun the job.

subtext	RECON DATA SET GET FAILED (RC=xx(XX) REASON CODE=yyy(YY))
----------------	--

Explanation

An attempt to GET a RECON record failed. The error return code and reason code are shown. The macro failed with return code xx in decimal format (XX in hexadecimal format) and reason code yyy in decimal format (YY as hexadecimal format).

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

To determine the cause of the problem, see *DFSMS Macro Instructions for Data Sets*, which explains the error return codes of the GET macro. Correct any errors, and rerun the job. If this situation persists, contact IBM Software Support.

subtext	OBTAIN FAILED FOR VOL=SER=aaaaaa (RC=xx)
----------------	---

Explanation

Program FABADA1 or FABCUR1 issued an OBTAIN macro to reserve volume *aaaaaa*. The attempt failed.

System action

FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Check the volume, fix the problem, and rerun the job.

subtext	INSUFFICIENT STORAGE FOR: aaaa - INCREASE REGION SIZE
----------------	--

Explanation

Program FABADA1 or FABCUR1 issued a GETMAIN macro to allocate storage for the purpose of *aaaa*. The attempt failed.

System action

FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Check the region size, increase the REGION parameter in the EXEC statement for FABADA1 or FABCUR1 as required, and rerun the job.

subtext	MINVERS LEVEL IS NOT CORRECT FOR RECON DATA SET
----------------	--

Explanation

The data set used for the DD name was not a correct IMS release level of the MINVERS mode RECON data set.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Specify the correct IMS release level of the MINVERS mode RECON data set, and rerun the job.

subtext	DYNALLOC FAILED (RC=rrrr RSN=eeeeiiii)
----------------	---

Explanation

DYNALLOC macro failed. Here, *rrrr* is the return code from SVC99, *eeee* is the SVC99 ERROR contents, and *iiii* is the SVC99 INFO contents. The return code and the reason code are described in the *MVS Programming: Authorized Assembler Services Reference*.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Correct the error and rerun the job.

subtext	DYNALLOC FAILED (DFSMDA MEMBER LOAD FAILED)
----------------	--

Explanation

The DFSMDA member was not found for the RECON data set.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Correct the error and rerun the job.

subtext	DYNALLOC FAILED (INCORRECT DFSMDA MEMBER)
----------------	--

Explanation

Member *ddname1/2/3* was loaded as a DFSMDA member, but it does not have the correct DFSMDA format. The eye catcher 'MDA' is not found in the member.

System action

FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Correct the error and rerun the job.

subtext	DYNALLOC FAILED (BLDL FAILED (RC=rrrr))
----------------	--

Explanation

Program FABADA1 or FABCUR1 issued a BLDL macro for the RECON data set *ddname1/2/3* specified in JOBLIB, STEPLIB, or SYSLIB. The macro failed. Here, *rrrr* is the return code from the macro. The return code is described in *DFSMS Macro Instructions for Data Sets*.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Correct the error and rerun the job.

subtext	DYNAMIC DEALLOCATION FAILED (RC=rrrr RSN=eeeeiiii)
----------------	---

Explanation

An attempt for dynamic deallocation of the *ddname* failed. Here, *rrrr* is the return code from SVC99, *eeee* is the SVC99 ERROR contents, and *iiii* is the SVC99 INFO contents. The return code and the reason code are described in the *MVS Programming: Authorized Assembler Services Reference*.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Correct the error and rerun the job.

subtext	DUPLICATED
----------------	-------------------

Explanation

The same RECON data set is specified.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Correct the error and rerun the job.

subtext	USABLE AS COPYn
----------------	-------------------------------------

Explanation

This RECON data set was accepted as a COPY n ($n=1$ or 2), but the valid RECON data set used as a pair was not specified.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

User response

Correct the error and rerun the job.

subtext	NOT USED
----------------	-----------------

Explanation

These are the possible reasons for the error:

- The RECON DD is not specified.
- The RECON data set was specified as DUMMY or NULLFILE.
- The RECON data set was empty.
- This is a spare RECON data set.

System action

Program FABADA1 or FABCUR1 ends with an abend code of 3700.

HFPB messages

The following information is about messages and codes that begin with HFPB.

HFPB0001I	INDEXBLD SCAN PROCESSING STARTED FOR AREA NO: $nnnn$, AREANAME: $areaname$ - INPUT DATA SET IS IMAGE COPY.
------------------	---

Explanation

The scan phase of the Build Index process for the secondary index database has started. When the input data set is an image copy, message INPUT DATA SET IS IMAGE COPY is issued.

System action

Processing continues.

User response

None. This message is informational.

User response

Correct the error and rerun the job.

FABD3701E	ERROR IN "TERM" CALL TO FABCRMIF - FUNC="TERM"; "DBDNAME" PARAMETER NOT FOUND - TERMINATION HAS BEEN PERFORMED ALREADY OR NOT FOUND FOR DMB MEMBER XXXXXXXX
------------------	--

Explanation

Program FABCRMIF/FABDRMIO determined that the parameter list specified by the calling program was incorrect.

System action

FABCRMIF/FABDRMIO ends with an abend code of 4011.

User response

Correct the CALL specifications in the program being used to invoke FABCRMIF/FABDRMIO. Then rerun the job.

HFPB0002I	INDEXBLD SCAN PROCESSING COMPLETED FOR AREA NO: $nnnn$, AREANAME: $areaname$ (ELAPSED TIME: $hh:mm:ss.tt$) - THE AREA WAS NOT SCANNED BECAUSE THE AREA IS EMPTY. - THE AREA WAS SCANNED. THE AREA IS EMPTY.
------------------	---

Explanation

The scan phase of the Build Index process for the secondary index database has completed. ELAPSED TIME shows the time spent for the processing.

- THE AREA WAS NOT SCANNED BECAUSE THE AREA IS EMPTY.

The Build Index function did not scan the area because the area is flagged as empty.

- THE AREA WAS SCANNED. THE AREA IS EMPTY.

The Build Index function scanned the area because the area is not flagged as empty. But the area contained no segments.

System action

Processing continues.

User response

None. This message is informational.

**HFPB0003I INDEXBLD [LOAD | COMPARE
| UPDATE | FS_RECLAIM]
PROCESSING STARTED FOR
SECONDARY INDEX DBD:
index_dbdname**

Explanation

The load phase, the compare phase, the update phase, or the freespace reclaim phase of the Build Index process for the indicated secondary index database started.

System action

Processing continues.

User response

None. This message is informational.

**HFPB0004I INDEXBLD [LOAD | COMPARE
| UPDATE | FS_RECLAIM]
PROCESSING COMPLETED FOR
SECONDARY INDEX DBD:
index_dbdname (ELAPSED TIME:
hh:mm:ss.tt)
- THE SECONDARY INDEX
DATABASE DID NOT NEED TO
BE UPDATED. ALREADY IN SYNC
WITH THE DEDB.**

Explanation

The load phase, the compare phase, the update phase, or the freespace reclaim phase of the Build Index process for the indicated secondary index database completed. ELAPSED TIME shows the time spent for the processing. When the secondary index database is already in sync with the DEDB, the subtext is printed.

System action

Processing continues.

User response

None. This message is informational.

**HFPB0005I DSNNAME FOR ddname DD IS NOT
REGISTERED TO DBRC.**

Explanation

The specified secondary index database is not registered to DBRC.

System action

Processing continues.

User response

None. This message is informational.

**HFPB0006W RESYNC POINTER SEGMENT
DUMP PROCESSING FOR
SECONDARY INDEX DATABASE:
index_dbdname WAS SKIPPED.
- OPEN FAILED FOR DDNAME:
ddname**

Explanation

Failed to open the data set that is indicated by ddname. This DD points to the resync pointer segment record data set for the indicated secondary index database.

System action

The program skips processing of the indicated secondary index, sets the end-of-job return code to 4, and continues processing the next secondary index database.

User response

Correct the errors, and rerun the job. If this situation persists, contact IBM Software Support.

**HFPB0007E IMS IS NOT AT THE
REQUIRED MAINTENANCE LEVEL
TO SUPPORT [TOICTL=NONE |
IDXPROC=FS_RECLAIM].**

Explanation

The IMS subsystem is not at the required maintenance level to support the TOICTL=NONE option or the IDXPROC=FS_RECLAIM option.

System action

FPA issues message HFPT3309E and ends the job with an abend code of U3309.

User response

To activate the TOICTL=NONE option, apply the PTF for APAR PI39873 to IMS 14.

To activate the IDXPROC=FS_RECLAIM option, apply the PTF for APAR PI62621 to IMS 14.

HFPB0008E	OPEN FAILED IN THE INDEX RESYNC PROCESS. DDNAME: <i>ddname</i>
------------------	--

Explanation

FPA could not open the data set that the indicated DD statement points to.

System action

FPA issues message HFPT3309E and ends the job with an abend code of U3309.

User response

This error is likely an internal error. Contact IBM Software Support.

HFPB0009E	GETMAIN FAILED IN THE INDEX RESYNC PROCESS.
------------------	--

Explanation

GETMAIN failed while processing the DL/I task.

System action

FPA issues message HFPT3309E and ends the job with an abend code of U3309.

User response

If the region size that is specified is too small, increase the REGION size on the JOB statement in the JCL. Then rerun the utility.

HFPB0010E	A POINTER SEGMENT FOR DLET WAS NOT FOUND. SECONDARY INDEX DATABASE: <i>index_dbdname</i>
------------------	---

Explanation

A pointer segment to be deleted was not found in the indicated secondary index database.

System action

The program skips processing of the indicated secondary index, sets the end-of-job return code to 8, and continues processing the next secondary index database.

User response

If RESYNCDUMP=YES is specified, identify the failed segment in the Resync Pointer Segment Dump report. Correct the errors, and rerun the job.

HFPB0011E	INCORRECT RESYNC POINTER SEGMENT RECORD DATA SET IS SPECIFIED. - SPECIFIED RESYNC POINTER SEGMENT RECORD DATA SET NAME: <i>dsname</i> - THE HEADER RECORD MUST CONTAIN THE FOLLOWING SECONDARY INDEX DATABASE NAME: <i>index_dbdname</i> - THE HEADER RECORD MUST CONTAIN THE FOLLOWING RESYNC POINTER SEGMENT RECORD DATA SET DDNAME: <i>S0nnnnnnR</i> - THE RECORD TYPE IS NOT OF A RESYNC POINTER SEGMENT RECORD.
------------------	---

Explanation

An incorrect resync pointer segment record data set is specified as input. To run the job in update mode (RESYNCMODE=UPDATE), a valid resync pointer segment record data set must be provided.

System action

The program skips processing of the indicated secondary index, sets the end-of-job return code to 8, and continues processing the next secondary index database.

User response

Specify the correct resync pointer segment record data set and rerun the job.

HFPB0012E	DUPLICATE KEYS FOUND IN SECONDARY INDEX DATABASE <i>dbdname</i>
------------------	--

Explanation

One or more duplicate keys were found in the indicated secondary index database during the Build Index process.

System action

The return code is set to 8 and processing continues.

User response

Duplicate keys are reported in the Secondary Index Duplicate Key report. Identify the duplicate keys from the report and remove the records with duplicate keys. Then, rerun the job. To interpret the report, see the topic "Secondary Index Duplicate Key report" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

HFPB0013E	NUMBER OF DUPLICATE KEYS REACHED THE LIMIT DEFINED BY DUPKEYMAX. SECONDARY INDEX DATABASE: <i>dbdname</i>
------------------	--

Explanation

The number of duplicate keys that were found in the indicated secondary index database reached the maximum allowable number of duplicate keys that is defined by the DUPKEYMAX keyword.

System action

If PROC_AFT_DUPKEY=CONT is specified, FPA sets the return code to 8 and continues processing the remaining secondary index databases. If PROC_AFT_DUPKEY=STOP is specified, FPA ends the job with a return code of 8.

User response

Identify the cause of the key duplication. Correct the errors and rerun the job. For more information about duplicate keys, see the topic "Secondary Index Duplicate Key report" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

HFPB0014I	PROC_AFT_DUPKEY=STOP IS SPECIFIED. INDEXBLD PROCESSING IS CANCELED FOR ALL SECONDARY INDEX DATABASES.
------------------	--

Explanation

PROC_AFT_DUPKEY=STOP is specified and the number of duplicate keys in a secondary index

database reached the maximum allowable number of duplicate keys that is defined by the DUPKEYMAX keyword. The Build Index function or the Resync function cancels the processing for the remaining secondary index databases.

System action

Processing continues.

User response

None. This message is informational.

HFPB0015E	SECONDARY INDEX DUPLICATE KEY REPORT FOR <i>dbdname</i> IS NOT GENERATED BECAUSE OF THE FOLLOWING REASON: - DYNAMIC ALLOCATION FAILED FOR DDNAME <i>ddname</i>, RC=<i>rc</i>, RSN=<i>rsn</i>
------------------	---

Explanation

The Build Index function or the Resync function could not obtain the information that is required to generate the Secondary Index Duplicate Key report for the indicated secondary index database. The reason is described in the subtext.

System action

The Build Index function or the Resync function does not generate the Secondary Index Duplicate Key report for the indicated secondary index database. Processing continues.

User response

Look up the dynamic allocation code (SVC99) in the *MVS Programming: Assembler Services Reference*. Correct the problem, then rerun the job.

HFPB0016I	INDEXBLD FS_RECLAIM PROCESSING SKIPPED FOR SECONDARY INDEX DBD: <i>index_dbdname</i> - THE DATABASE ORGANIZATION TYPE IS HISAM AND THE OVERFLOW DATA SET IS DEFINED. - THE DATABASE ORGANIZATION TYPE IS SHISAM.
------------------	---

Explanation

The freespace reclaim phase of the Build Index process for the indicated secondary index database was skipped. Subtext shows the reason why the resource was skipped.

System action

Processing continues.

User response

None. This message is informational.

HFPB3500E **AN ERROR WAS RETURNED FROM USER PARTITION SELECTION EXIT ROUTINE:**
routine, FUNCTION: *function*, RC: *rc*
- TARGET SEGMENT: *segment*, XDFLD: *xdfld*
- AREANAME: *areaname*, SOURCE SEGMENT: *source_segment*, RBA: *nnnnnnnn*

Explanation

An error was returned from the indicated user partition selection exit routine.

System action

The job ends with an abend code of U3003.

User response

Check the indicated user partition selection exit, the function, and the return code. Correct the errors and rerun the job.

HFPB3501E **THE DBDNAME RETURNED FROM USER PARTITION SELECTION EXIT ROUTINE:** *routine* **IS NOT FOUND IN THE DBD DEFINITION.**
- AREA NO: *areanum*, AREANAME: *areaname*, SECONDARY INDEX DBD: *dbdname*, XDLFD: *xdfld*
- TARGET SEGMENT: *target_segment*, SOURCE SEGMENT: *source_segment*

Explanation

The *dbdname* that is returned from the indicated user partition selection exit is not found in the primary DEDB DBD definition.

System action

The job ends with an abend code of U3003.

User response

Correct the errors and rerun the job.

HFPB3510E **KEYLEN VALUE IN VSAM CATALOG DIFFERS FROM DBD DEFINITION. SECONDARY INDEX DATABASE:** *database*, DDNAME: *ddname*

Explanation

The KEYLEN value in the VSAM catalog is not the same as the corresponding value in the DBD for the secondary index database data set.

System action

The job ends with an abend code of U3003.

User response

Correct the error and rerun the job.

HFPB3511E **THE RECORD SIZE IS TOO SMALL TO STORE THE POINTER SEGMENT. SECONDARY INDEX DATABASE:** *database*, DDNAME: *ddname*

Explanation

The record size is too small to store the pointer segment.

System action

The job ends with an abend code of U3003.

User response

Correct the error and rerun the job.

HFPB3512E **CATALOG INFORMATION COULD NOT BE OBTAINED. SECONDARY INDEX DATABASE:** *database*, DDNAME: *ddname*

Explanation

Catalog information could not be obtained.

System action

The job ends with an abend code of U3003.

User response

Correct the error and rerun the job.

HFPB3513E **SECONDARY INDEX DBDS IS NOT EMPTY. SECONDARY INDEX DATABASE:** *database*, DDNAME: *ddname*

Explanation

Secondary index DBDS is not empty.

System action

The job ends with an abend code of U3003.

User response

Prepare an empty DBDS for the secondary index database and rerun the job.

HFPB3514E	<i>function</i> CALL FOR SECONDARY INDEX DATABASE FAILED. SECONDARY INDEX DATABASE: <i>database</i>, DDNAME: <i>ddname</i> - RETURN CODE IS <i>rc</i> (X'<i>hex_rc</i>'). RPL FEEDBACK AREA IS <i>rsn</i> (X'<i>hex_rsn</i>'). - RETURN CODE IS <i>rc</i> (X'<i>hex_rc</i>'). ACB ERROR FLAG CODE IS <i>rsn</i> (X'<i>hex_rsn</i>').
------------------	---

Explanation

The utility received a nonzero return code from VSAM when it attempted to access the VSAM data set for the secondary index database. *function* indicates the name of macro that was issued to access the VSAM data set. The return code and the reason code are shown both in decimal (*rc*, *rsn*) and hexadecimal (*hex_rc*, *hex_rsn*) formats.

System action

The job ends with an abend code of U3003.

User response

See *DFSMS Macro Instructions for Data Sets*, which describes VSAM administration macros. If this situation persists, contact IBM Software Support.

HFPB3515E	SYNAD EXIT WAS INVOKED FOR A VSAM DATA SET. SECONDARY INDEX DATABASE: <i>database</i>, DDNAME: <i>ddname</i>
------------------	---

Explanation

The SYNAD exit was invoked for the VSAM data set.

System action

The job ends with an abend code of U3003.

User response

Contact IBM Software Support.

HFPB3516E	OVERFLOW RECORD SIZE IS SMALLER THAN PRIME RECORD SIZE. SECONDARY INDEX DATABASE: <i>database</i>, DDNAME: <i>ddname</i>
------------------	---

Explanation

The overflow record size is smaller than the prime record size.

System action

The job ends with an abend code of U3003.

User response

Correct the error and rerun the job.

HFPB3517E	DATA SIZE EXCEEDED THE TOTAL SIZE OF ALLOCATED STRIPE FILES. <i>nnnnnnnnnn</i>K BYTES USED IN DD <i>Snnnnnn</i>0
------------------	---

Explanation

The data size has exceeded the size of the allocated stripe file.

System action

The job ends with an abend code of U3003.

User response

Increase the size of the indicated DD or specify the indicated DD with a larger size, and rerun the job.

HFPB3518E	FOUND A DUPLICATE KEY IN POINTER SEGMENT. OVERFLOW DATA SET MUST BE DEFINED. SECONDARY INDEX DATABASE: <i>dbdname</i>
------------------	--

Explanation

A duplicate key was found in a pointer segment. However, an overflow data set was not defined.

System action

The job ends with an abend code of U3003.

User response

Correct the error and rerun the job.

**HFPB3519E THE RECORD SIZE IS ODD-SIZE.
SECONDARY INDEX DATABASE:
database, DDNAME: ddname**

Explanation

The record has an odd size. The size must be an even.

System action

The job ends with an abend code of U3003.

User response

Correct the error and rerun the job.

**HFPB3520E KEY OFFSET IN VSAM CATALOG
DIFFERS FROM DBD DEFINITION.
SECONDARY INDEX DATABASE:
database, DDNAME: ddname**

Explanation

The KEY OFFSET value in the VSAM catalog is not the same as the corresponding value in the DBD for the secondary index database data set.

System action

The job ends with an abend code of U3003.

User response

Correct the error and rerun the job.

**HFPB3521E ERROR IN CALL TO RANDOMIZER:
rmodname WHILE ANALYZING
SECONDARY INDEX DATABASE:
database_name
- RETURN CODE: nnnn**

Explanation

The Analyze function invoked the randomizer routine that was specified to calculate the area number and the RAP RBA information. The return code from the randomizer routine shown in the error message was not zero.

System action

The job ends with an abend code of U3003.

User response

Make sure that the RMODLIB DD statement is identified in the correct data set, and that the randomizer routine has been correctly added,

assembled, and link-edited. Correct any errors and rerun the job.

**HFPB3522E INCORRECT BUILD POINTER
SEGMENT RECORD DATA SET IS
SPECIFIED. DDNAME: S0nnnnnn0
- HEADER RECORD IS NOT FOUND.
- INVALID RECORD TYPE IS
FOUND.
- THE HEADER RECORD MUST
CONTAIN THE FOLLOWING
SECONDARY INDEX DATABASE
NAME: indexdbd
- THE HEADER RECORD MUST
CONTAIN THE FOLLOWING BUILD
POINTER SEGMENT RECORD DATA
SET DDNAME: S0nnnnnn0
- RECORD LENGTH IS TOO SHORT
(xxxxx BYTES SPECIFIED, xxxxx
BYTES REQUIRED)**

Explanation

An incorrect build pointer segment record data set is specified as input. To run the job in load mode (BUILD MODE=LOAD), a valid build pointer segment record data set must be provided.

System action

The job ends with an abend code of U3003.

User response

Specify the correct build pointer segment record data set and rerun the job.

**HFPB3523E [BUILD | UNVERIFIED RESYNC]
POINTER SEGMENT RECORDS ARE
NOT SORTED. DDNAME: S0nnnnnn0**

Explanation

The supplied pointer segment record data set contains unsorted records. When the Build Index process runs with one of the following keyword-parameter combinations, the supplied record data sets must be sorted in advance.

- IDXPROC=BUILD, BUILD MODE=LOAD, and SORT=NO
- IDXPROC=RESYNC, RESYNCMODE=VERIFYUPDATE, and AREASCAN=NO

System action

The job ends with an abend code of U3003.

User response

Sort the supplied pointer segment record data set, and then rerun the job.

**- THE DATA SET CONTAINS
DUPLICATE RECORDS FOR AREA
NUMBER *nnnn***

HFPB3524E **INCORRECT UNVERIFIED RESYNC
POINTER SEGMENT RECORD DATA
SET IS SPECIFIED.**
**- SPECIFIED UNVERIFIED RESYNC
POINTER SEGMENT RECORD DATA
SET NAME: *dsname***
**- THE HEADER RECORD MUST
CONTAIN THE FOLLOWING
SECONDARY INDEX DATABASE
NAME: *index_dbdname***
**- THE HEADER RECORD MUST
CONTAIN THE FOLLOWING
UNVERIFIED RESYNC POINTER
SEGMENT RECORD DATA SET
DDNAME: *S0nnnnn0***
**- THE RECORD TYPE IS NOT OF AN
UNVERIFIED RESYNC POINTER
SEGMENT RECORD.**
**- THE AREA NUMBER *nnnn* IS
INCORRECT.**

Explanation

An incorrect unverified resync pointer segment record data set is specified as input. To run the job in verify update mode (RESYNCMODE=VERIFYUPDATE) or direct update mode (RESYNCMODE=DIRECTUPDATE) with the AREASCAN=NO option, a valid unverified resync pointer segment record data set must be provided.

System action

The job ends with an abend code of U3003.

User response

Specify the correct unverified resync pointer segment record data set and rerun the job.

Gathering diagnostic information

Before you report a problem with IMS HP Fast Path Utilities to IBM Software Support, you need to gather the appropriate diagnostic information.

Provide the following information for all supplementary utility problems:

- 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
- Snap dump generated in the HFPABEND data set

The HFPABEND data set is generated only when the FPA process ends abnormally. If the HFPABEND DD is not specified in the JCL, FPA dynamically allocates the data set by using SYSOUT=*.

- A Load Module/Macro APAR Status report

Use the Diagnostics aid to create a Load Module/Macro APAR Status report. For more information, see the topic "Diagnostics aid" in the *IMS Fast Path Solution Pack: IMS High Performance Fast Path Utilities User's Guide*.

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.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Programming interface information

This information is intended to help the customer use the functional capabilities of IMS High Performance Fast Path Utilities of IBM IMS Fast Path Solution Pack for z/OS.

However, this information also documents Product-Sensitive programming interface information and associated guidance information.

Product-Sensitive programming interfaces are provided to allow the customer installation to perform tasks such as tailoring, monitoring, modification, or diagnosis of this IBM product. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM product. Product-Sensitive interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-Sensitive programming interface information is explicitly identified where it occurs, as an introductory statement to a chapter or section that is entirely Product-Sensitive programming interface information.

Product-Sensitive Programming Interface and Associated Guidance Information is identified where it occurs, either as an introductory statement to a chapter or section or by the following marking:



Product-sensitive programming interface and associated guidance information...



Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com)® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>.

Other company, product, and service names may be trademarks or service marks of others.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions:

Applicability: These terms and conditions are in addition to any terms of use for the IBM website.

Personal use: You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights: Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

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

- accessibility
 - overview [5](#)
- alias for FABCUR7 [54](#)
- application interface
 - DEDB Reload Segment Data Set Create utility (FABCUR6) [37](#)
- application interface, DEDB Reload Segment Data Set Create utility (FABCUR6)
 - EOF function [37](#)
 - INIT function [37](#)
- application interface, FABCUR6
 - PUT function [37](#)

B

- build and write segment, PUT [37](#)
- BYPASS keyword, DEDB/HD Unload Conversion utility [95](#)

C

- CHKP control statement [86](#)
- CNTLCRDS control statements [86](#)
- COBOL
 - coding EOF function [37](#), [55](#)
 - coding GET/GET1/GET2 [55](#)
 - coding INIT [37](#)
 - coding INIT/INID [55](#)
 - coding PUT [37](#)
- control statement syntax
 - FABCUR5 [28](#)
- control statements
 - DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) [61](#)
- conversion, database
 - DEDB/HD Unload Conversion utility (FABCUR9) [83](#)
- cookie policy [247](#)
- creating a DEDB database from an HD unload file
 - DEDB/HD Unload Conversion utility (FABCUR9) [84](#)
- creating a HIDAM database from an HDAM unload file
 - DEDB/HD Unload Conversion utility (FABCUR9) [84](#)
- creating an HD database from a DEDB unload file
 - DEDB/HD Unload Conversion utility (FABCUR9) [83](#)

D

- data and system flow
 - Database Definition Record Create utility (FABCUR5) [25](#)
 - DEDB Reload Segment Data Set Create utility (FABCUR6) [35](#)
 - DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) [53](#)
 - DEDB/HD Unload Conversion utility (FABCUR9) [82](#)
 - HD To DEDB Unload Data Set Conversion utility (FABCUR8) [70](#)

- data sets
 - SDEP Data Collection [10](#)
 - SDEP History [10](#)
- database conversion
 - DEDB/HD Unload Conversion utility (FABCUR9) [83](#)
- Database Definition Record Create utility (FABCUR5)
 - data and system flow [25](#)
 - DD statements [26](#)
 - example [33](#)
 - input [28](#)
 - JCL
 - sample [25](#)
 - job control language (JCL) [26](#)
 - output [29](#)
 - overview [25](#)
- DBDNAME control statement [86](#)
- DD statements
 - Database Definition Record Create utility (FABCUR5) [26](#)
 - DEDB Reload Segment Data Set Create utility (FABCUR6) [39](#)
 - DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) [58](#)
 - DEDB/HD Unload Conversion utility (FABCUR9) [85](#)
 - DFSORT (step SORTSDEP) [13](#)
 - FABADA7 [12](#)
 - FABADA8 [13](#)
 - FABADA9 [14](#)
 - HD To DEDB Unload Data Set Conversion utility (FABCUR8) [71](#)
- DEDB Pointer Checker
 - messages and codes [140](#)
- DEDB Reload Segment Data Set Create utility (FABCUR6)
 - application interface
 - EOF function [37](#)
 - INIT function [37](#)
 - PUT function [37](#)
 - data and system flow [35](#)
 - DD statements [39](#)
 - dynamic linkage [36](#)
 - example [50](#)
 - input [42](#)
 - job control language
 - sample [36](#)
 - output [46](#)
 - overview [35](#)
 - site default [48](#)
 - static linkage [37](#)
 - UR6CTL data set [42](#)
 - using [36](#)
- DEDB Unload/Reload
 - RBA conversion [111](#)
- DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7)
 - application interface
 - EOF function [55](#)
 - GET/GET1/GET2 functions [55](#)

DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) (continued)

application interface (continued)

INIT/INID function [55](#)

data and system flow [53](#)

DD statements [58](#)

dynamic linkage [54](#)

example [65](#)

functions [69](#)

input

DEDB segment data [61](#)

job control language (JCL) [58](#)

output [63](#), [64](#)

overview [53](#)

static linkage [54](#)

UR7CTL control statements

AREA_INFORMATION_RECORD [61](#)

DBDNAME [61](#)

EXITRTN [61](#)

IMSCATACB_INPUT [63](#)

IMSCATHLQ [63](#)

IMSCOMP [61](#)

UR7CTL data set [61](#)

using [54](#)

DEDB/HD Unload Conversion utility (FABCUR9)

a DEDB database from an HD unload file [84](#)

control statements

syntax [86](#)

creating a HIDAM database from an HDAM unload file [84](#)

creating an HD database from a DEDB unload file [83](#)

data and system flow [82](#)

database conversion [83](#)

DD statements [85](#)

example [98](#)

functions [81](#)

input [86](#)

output

Summary report [95](#)

overview [81](#)

PSB requirements [85](#)

recovery/restart [84](#)

restrictions [82](#)

segment cross-reference keywords [89](#)

segment cross-reference records [89](#)

segment cross-reference table [89](#)

site default [97](#)

using [83](#)

DEDB/HD Unload Conversion Utility (FABCUR9) [95](#)

DFSORT (SOTRSDEP) exec statement [14](#)

DFSORT control statement [16](#)

DFSORT JCL

SORTSDEP [13](#)

documentation

accessing [3](#)

sending feedback [3](#)

dynamic linkage

DEDB Reload Segment Data Set Create utility

(FABCUR6) [36](#)

DEDB Unloaded Segment Data Set Retrieve utility

(FABCUR7) [54](#)

E

EOF

function [37](#), [55](#)

examples

Database Definition Record Create utility (FABCUR5) [33](#)

DEDB Reload Segment Data Set Create utility

(FABCUR6) [50](#)

DEDB Unloaded Segment Data Set Retrieve utility

(FABCUR7) [65](#)

DEDB/HD Unload Conversion utility (FABCUR9)

Segment Cross-Reference files for segment format

conversions [107](#)

using an HD unload file [100](#)

using the database definition record (DURDBDFN)

[99](#)

using the Segment Cross-Reference table [101](#)

HD To DEDB Unload Data Set Conversion utility

(FABCUR8) [79](#)

SDEP Space Utilization utility

extracting SDEP space utilization data [21](#)

IOVF processing [20](#)

SDEP part processing [20](#)

SDEP update only [22](#)

SDEP Utilization reports only [22](#)

update and report [20](#)

extended root segment information area layout [56](#), [57](#)

F

FABADA7

JCL [12](#)

SYSIN data set [15](#)

FABADA8

JCL [13](#)

SYSIN data set [16](#)

SYSPRINT data set [17](#)

FABADA8 exec statement [13](#)

FABADA9

DD statements [14](#)

JCL [14](#)

MSGOUT data set [19](#)

RPTOUT data set [17](#)

SYSIN data set [16](#)

FABADA9 exec statement [14](#)

FABADA9 SYSIN keywords

AREA [17](#)

DBDNAME [17](#)

FABC messages [149](#)

FABCP6J [48](#)

FABCP6M [48](#)

FABCP9J [97](#)

FABCP9M [97](#)

FABCRMIF [111](#)

FABCUR5

control statements [28](#)

FABCUR5 exec statement [26](#)

FABCUR5 SYSIN control statements

DBDNAME [28](#)

FUNCTION [28](#)

FABCUR6

control statements [42](#), [74](#)

input [42](#), [74](#)

job control language (JCL) [39](#)

FABCUR6 UR6CTL control statements

AREA_INFORMATION_RECORD [42](#), [74](#)

FABCUR6 UR6CTL control statements (*continued*)

- EXITRTN [42, 74](#)
- FILECTL [42, 74](#)
- FORMAT [42, 74](#)
- IMSCOMP [42, 74](#)
- LRECL [42, 74](#)
- OUTDD [42, 74](#)
- USERCTL [42, 74](#)

FABCUR7

- alias FABEUR7 [54](#)
- COBOL coding [55](#)
- EOF function [55](#)
- GET/GET1/GET2 functions [55](#)

FABCUR8

- alias FABEUR8 [70](#)
- exec statement [71](#)
- FABCUR8
exec statement parameter [71](#)

FABCUR9

- output [95](#)

FABCUR9 CNTLCRDS control statements

- CHKP [86](#)
- DBDNAME [86](#)
- GHU [86](#)
- INPUT [86](#)
- OUTFILE [89](#)
- REPL [89](#)
- SEGXREFI [89](#)
- SEGXREFO [89](#)
- TEST [89](#)

FABEUR7, alias for FABCUR7 [54](#)

FABGXDR [117](#)

FPXGXDR0 [117](#)

functions

- Database Definition Record Create utility (FABCUR5) [25](#)
- DEDB Reload Segment Data Set Create utility (FABCUR6) [35](#)
- DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) [53, 69](#)
- DEDB/HD Unload Conversion utility (FABCUR9) [81](#)
- SDEP Space Utilization utility [7](#)

G

GET/GET1/GET2 functions [55](#)

GET/GET1/GET2, coding in COBOL [55](#)

GHU keyword, DEDB/HD Unload Conversion utility (FABCUR9) [86, 94](#)

H

HD To DEDB Unload Data Set Conversion utility (FABCUR8)

- data and system flow [70](#)
- DD statements [71](#)
- example [79](#)
- input [74](#)
- output [78](#)
- overview [69](#)
- UR6CTL data set [74](#)
- using [70](#)

HDAM/HIDAM database conversion [83](#)

I

IMS Database Recovery Facility [135](#)

IMS HP Image Copy [127](#)

IMS management of ACBs [3](#)

IMS-managed ACBs [3](#)

INIT

- coding in a COBOL program [37](#)

- function [37](#)

INIT/INID in a COBOL program [55](#)

initialize permanent data sets [10](#)

input

- control statement syntax [16](#)

- Database Definition Record Create utility (FABCUR5) [28](#)

- DEDB Reload Segment Data Set Create utility

 - (FABCUR6)

 - UR6CTL data set [42](#)

- DEDB Unloaded Segment Data Set Retrieve utility

 - (FABCUR7)

 - UR7CTL data set [61](#)

- DEDB/HD Unload Conversion utility (FABCUR9)

 - CNTLCRDS data set [86](#)

 - segment cross-reference records [89](#)

 - SEGREFI data set [89](#)

- FABADA7 SYSIN data set [15](#)

- FABADA8 utility control statements [16](#)

- FABADA9 SYSIN data set [16](#)

- FABCUR5 SYSIN data set [28](#)

- FABCUR6 [42, 74](#)

- HD To DEDB Unload Data Set Conversion utility

 - (FABCUR8)

 - UR6CTL data set [74](#)

- SDEP Space Utilization utility [15](#)

- SORTSDEP SYSIN data set [16](#)

- UR7DATA, UR7DATA1, and UR7DATA2 data sets [61](#)

INPUT control statement [86](#)

input data sets

- standard format extract data interface [117](#)

interface parameter

- initialization parameter [112](#)

- invocation parameter [112](#)

invoking a randomizing module [111](#)

J

job control language (JCL)

- Database Definition Record Create utility (FABCUR5) [26](#)

- DEDB Reload Segment Data Set Create utility

 - (FABCUR6) [39](#)

- DEDB Unloaded Segment Data Set Retrieve utility

 - (FABCUR7) [58](#)

- DFSORT (step SORTSDEP) [13](#)

- FABADA7 [12](#)

- FABADA8 [13](#)

- FABADA9 [14](#)

- FABARMIF [114](#)

procedures

- SDEP Space Utilization utility [10](#)

- randomizing module [114](#)

- SDEP Space Utilization utility [11](#)

- standard format extract data interface [119](#)

- to initialize permanent data sets [10](#)

job steps

- SDEP Space Utilization utility [7, 11](#)

K

keyboard shortcuts [5](#)

L

legal notices

cookie policy [247](#)

notices [247](#)

programming interface information [247](#)

trademarks [247](#)

link-edit for randomizing module [114](#)

load modules

SDEP Space Utilization utility [7](#)

M

messages

FPA

Build Index process and Analyze (index verify)

process [239](#)

FPB

common routines [230](#)

DEDB Pointer Checker [140](#)

OSM [139](#)

MSGOUT data set

FABADA9 [17](#), [19](#)

multiple input file mode [117](#)

N

NOREP keyword, DEDB/HD Unload Conversion utility [95](#)

notices [247](#)

O

OUTFILE keyword, DEDB/HD Unload Conversion utility (FABCUR9) [89](#)

output

Database Definition Record Create utility (FABCUR5) [29](#)

DEDB Reload Segment Data Set Create utility (FABCUR6)

UR6AUDIT data set [47](#)

UR6PRINT data set [46](#)

DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7)

UR7AUDIT data set [64](#)

UR7PRINT data set [63](#)

DEDB/HD Unload Conversion utility (FABCUR9)

SYSPRINT data set [95](#)

FABADA8 SYSPRINT data set [17](#)

FABADA9 MSGOUT data set [19](#)

FABADA9 RPTOUT data set [17](#)

FABCUR5 REPORTS data set [29](#)

FABCUR5 SYSPRINT data set [29](#)

FABCUR9 [95](#)

HD To DEDB Unload Data Set Conversion utility (FABCUR8)

UR6AUDIT data set [78](#)

UR6PRINT data set [78](#)

output files

standard format extract data interface [119](#)

output report, standard format extract [126](#)

overview

supplementary utilities [1](#)

P

parameter list, standard format extract data interface [118](#)

permanent data sets

SDEP Data Collection data set [10](#)

SDEP History data set [10](#)

prerequisite knowledge [3](#)

prerequisite publications [3](#)

programming interface information [247](#)

PSB requirements, DEDB/HD Unload Conversion utility (FABCUR9) [85](#)

PUT

coding in a COBOL program [37](#)

function [37](#)

R

randomizing module [111](#)

randomizing module, invoking

initialization parameter [112](#)

input [115](#)

interface parameter [112](#)

invocation parameter [112](#)

JCL required [114](#)

link-editing [114](#)

module names for [111](#)

RMIFCTL [115](#)

RBA conversion (DEDB Unload/Reload) [111](#)

reader comment form [3](#)

recovery/restart [84](#)

REPL keyword, DEDB/HD Unload Conversion utility (FABCUR9) [89](#), [95](#)

reports

Database Definition Record Create utility (FABCUR5)

DBD Definition report [29](#)

FABCUR5-Messages report [29](#)

DEDB Reload Segment Data Set Create utility (FABCUR6)

FABCUR6 Audit Control report [47](#)

FABCUR6-Messages report [46](#)

DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7)

FABCUR7-Messages report [63](#)

DEDB/HD Unload Conversion utility (FABCUR9)

Summary report [95](#)

HD To DEDB Unload Data Set Conversion utility (FABCUR8)

FABCUR6 Audit Control report [78](#)

FABCUR8-Messages report [78](#)

SDEP Space Utilization utility

SDEP Data Format-Messages [17](#)

SDEP History/Reports—Messages report [19](#)

SDEP Utilization report [17](#)

standard format extract [126](#)

restrictions

DEDB/HD Unload Conversion utility (FABCUR9) [82](#)

RPTOUT data set

FABADA9 [17](#)

S

- SDEP Data Collection data set [10](#)
- SDEP History data set [10](#)
- SDEP part processing
 - JCL [10](#)
 - steps [7](#)
- SDEP Space Utilization utility
 - examples [20](#)
- SDEP Utilization report, SDEP Space Utilization utility [7](#), [17](#)
- segment cross-reference records [89](#)
- segment I/O area layout [56](#)
- SEGXREFI keyword, DEDB/HD Unload Conversion utility (FABCUR9) [89](#)
- SEGXREFO keyword, DEDB/HD Unload Conversion utility (FABCUR9) [89](#)
- Sequential Dependent Scan utility
 - control statement [15](#)
- service information [3](#)
- single input file mode, standard format extract data interface [117](#)
- site default support [48](#), [97](#)
- SORTSDEP step [13](#)
- SORTSDEP SYSIN data set [16](#)
- standard format extract
 - output report [126](#)
- standard format extract data interface
 - DD DATA statements [119](#)
 - example JCL [119](#)
 - input file(s) [117](#)
 - JCL [119](#)
 - modes of operation [117](#)
 - multiple input files [117](#)
 - parameter list [118](#)
 - single input file mode [117](#)
 - STEPLIB statement [119](#)
 - XDRPRINT statement [119](#)
- static linkage
 - DEDB Reload Segment Data Set Create utility (FABCUR6) [37](#)
 - DEDB Unloaded Segment Data Set Retrieve utility (FABCUR7) [54](#)
- step SORTSDEP [13](#)
- STEPLIB statement
 - standard format extract data interface [119](#)
- summary of changes [2](#)
- supplementary utilities of IMS HP Fast Path Utilities [1](#)
- support information [3](#)
- syntax
 - control statement [16](#)
 - DEDB Pointer Checker [16](#)
 - FABADA9 [16](#)
- SYSIN data set
 - Database Definition Record Create utility (FABCUR5) [28](#)
 - FABADA7 [15](#)
 - FABADA8 [16](#)
 - FABADA9 [16](#)
 - SORTSDEP [16](#)
- SYSPRINT data set
 - FABADA8 [17](#)
 - FABADA9 [19](#)

T

- technotes [3](#)
- TEST keyword, DEDB/HD Unload Conversion utility (FABCUR9) [89](#)
- trademarks [247](#)

U

- using
 - DEDB/HD Unload Conversion utility (FABCUR9) [83](#)

X

- XCI randomizer [37](#), [111–113](#), [116](#)
- XDRPRINT output file [119](#), [126](#)



Product Number: 5698-FPP

SC27-9598-03

