Version 1 Release 1

IBM IMS Sequential Randomizer Generator for OS/390 User's Guide

IBM

SC27-0926-03

Version 1 Release 1

IBM IMS Sequential Randomizer Generator for OS/390 User's Guide

IBM

SC27-0926-03

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

Fourth Edition (February 2015)

This edition applies to Version 1 Release 1 of IBM IMS Sequential Randomizer Generator for OS/390 (program number 5655-E11) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC27-0926-02.

Contents

About this information

IBM® IMS™ Sequential Randomizer Generator for OS/390® (also referred to as IMS Sequential Randomizer Generator or SRG) creates a randomizing module to access database segments directly or sequentially.

These topics provide instructions for installing, configuring, and using IMS Sequential Randomizer Generator.

These topics are designed to help database administrators, system programmers, application programmers, and system operators perform these tasks:

- Understand the functions of IMS Sequential Randomizer Generator
- Operate IMS Sequential Randomizer Generator
- Diagnose and recover from IMS Sequential Randomizer Generator problems

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

Always check the IMS Tools Product Documentation page for the most current version of this information:

http://www.ibm.com/support/docview.wss?uid=swg27020942

Chapter 1. Introduction

IBM IMS Sequential Randomizer Generator for OS/390 (also referred to as IMS Sequential Randomizer Generator or SRG) creates a randomizing module to access database segments directly or sequentially.

Topics:

What's new in IMS Sequential Randomizer Generator

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.

| ### SC27-0926-03 (February 2015)

| Added information about how Sequential Randomizers work for DEDBs. See
| "How the SR works for a DEDB" on page 73.

SC27-0926-02

APAR PQ83931
The DBLOFFST parameter has been added to the ANALYZE control statement. You can use this parameter for HDAM and PHDAM to supersede the DBLEN parameter. For more information, see "ANALYZE control statement" on page 32.

APAR PK53644
SRG supports the XCI randomizer interface that was enhanced by APAR PK40256 for IMS Version 10. For more information, see "DEDB SR of SRTYPE=MULTI" on page 73.

APAR PM59902
The KEYRNG parameter has been added to the AREADEF control statement. This parameter supersedes the ARETRY parameter. For more information, see "AREADEF control statement" on page 48.

APAR PM62545

- The TNUMLIM and OPTMTIME parameters have been added to the ANALYZE and the AREADEF control statements. You can use these parameters to enhance the optimization of the equation table.
- The SIZECHK parameter has been added to the ANALYZE control statement. You can use this parameter to determine whether the size of the SR module exceeds 16 MB.

For more information, see "ANALYZE control statement" on page 32 and "AREADEF control statement" on page 48.

SC27-0926-01

SRG now supports IMS Version 8. This edition describes the new capability.

SC27-0926-00

IMS Sequential Randomizer Generator for OS/390 Release 1 is released.

The main changes from IMS Sequential DAM Optimizer of IMS System Utilities/Data Base Tools Version 2 (Program Number: 5685-093) are the following:

- Support for the generation of the sequential randomizer for a PHDAM partition
- Support for the generation of the reusable or reentrant sequential randomizer for HDAM and PHDAM

Program functions

IMS Sequential Randomizer Generator creates a sequential randomizer (SR) that enables users to access segments in a data entry database (DEDB), an HDAM database, or a partition of a partitioned HDAM (PHDAM) database, either directly or in the sequence of root segment keys. In other words, a DEDB, an HDAM, or a partition of a PHDAM can be accessed in the sequence of root segment keys without sacrificing its capability for efficient direct access.

Sections:
- "Creating a randomizer for sequential processing"
- "Creating a randomizer for nonsequential processing"
- "Producing statistical information" on page 4

Note: The term *CI* (control interval) used in these topics should be interpreted as block in the case of an OSAM data set.

Creating a randomizer for sequential processing

IMS Sequential Randomizer Generator analyzes the database keys, database record lengths, and control statements to create a randomizing module called *Sequential Randomizer*, which enables sequential processing of a DEDB, an HDAM database, or a partition of a PHDAM database.

Sequential Randomizer (also referred to as SR) is a randomizing module that calculates the physical positions of root segments in a database in a way that the physical sequence of root segments matches the logical sequence. The physical sequence is indicated by a relative block number (RBN) and a root anchor point (RAP) number for an HDAM or PHDAM database, and a DEDB area and a relative RAP number for a DEDB database.

Note: Logical sequence is the sequence of root segment keys.

Figure 1 on page 4 contains a simplified picture of how IMS stores root segments in an HDAM or PHDAM database by ISRT (insert) calls, when SR is being used:
1. Application program issues an ISRT call of a root segment.
2. IMS receives control, and calls the SR.
3. The SR calculates the control interval (CI) number and RAP number using the root segment key value, and passes them back to IMS.
4. IMS stores the root segment in the specified CI and RAP position.

As shown in Figure 1 on page 4, the SR stores root segments by the key sequence, so that the physical sequence of root segments matches the logical sequence. Therefore, by issuing successive GN (get-next) calls, you can retrieve root segments in the key sequence.

For the detailed description of how SRG generates SR, see Chapter 5, "Reference: How the Sequential Randomizer is generated," on page 95.

Creating a randomizer for nonsequential processing

SRG provides a function to generate a *nonsequential* randomizer. A nonsequential randomizer cannot be used to retrieve or insert segments in a key sequence, but it can reduce synonyms as with SR. This function provides the applications with a better distribution of keys.

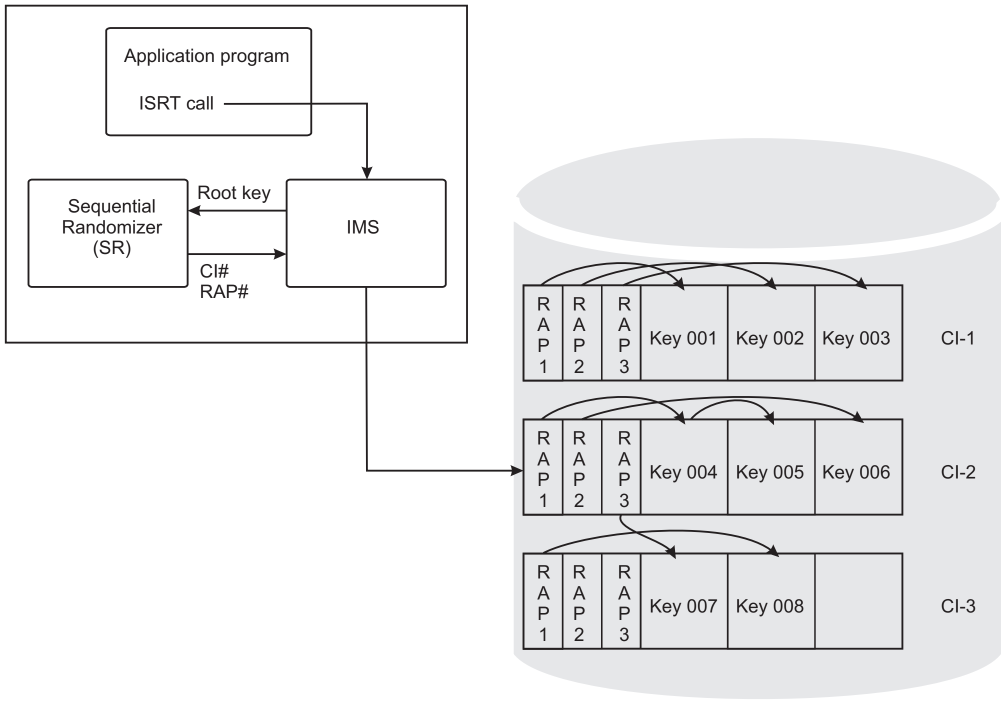


Figure 1. How IMS stores root segments in a database using SR (HDAM or PHDAM)

Producing statistical information

During the generation of a randomizing module, SRG simulates its execution. It calculates all the input database keys and their physical positions as if the randomizing module was invoked by IMS, and collects statistical data. Using this data, SRG produces various statistics reports to show the synonym distribution status and CI utilization, to help you evaluate the efficiency of the randomizing module and fine-tune it, if necessary.

SRG produces the following reports:

- Recommended DBD Control Statement report
- CI Distribution List report
- Randomizer Statistics report
- Synonym Distribution Status report
- DEDB Master Table report (DEDB only)

Typical benefits

Typical benefits from using SRG are as follows:

- Creates a randomizing module that can access DEDB, HDAM, or PHDAM segments either directly or sequentially, without sacrificing their efficient direct-access capability.
- Controls the number of synonyms of root segments in a database. If you know the attributes and distribution of the keys in advance, you can create a randomizing module that generates fewer synonyms.
- Controls the utilization ratio of control intervals (CIs) of the root addressable area in a database by considering the input database record length.
- Uses statistics provided on SRG reports to monitor the efficiency of the randomizing module.

Sequential Randomizer overview

Sequential Randomizer (SR) generated by SRG is composed of two parts; *Sequential Randomizing Routine* (also referred to as SRR), and *Sequential Randomizing Table* (also referred to as SRT).

SRR works as a randomizing program, which you normally generate and specify in the DBD. SRR calculates RAP numbers from the key values using a table (SRT), which contains linear equations.

An overview of SRG data flow is shown in Figure 2 on page 7.

1. SRG analyzes the SRG control statements that you specify. In the SRG control statements, you specify the characteristics of the database (HDAM, PHDAM, or DEDB), the characteristics of the input data set, and some options for creating a randomizer.
2. SRG reads the input database key file, which you create in advance. (*Database key file* is a data set that contains the key field of the root segment and the database record length field for each database record in a database.) Then, SRG compresses the database keys to improve the efficiency of database key analysis.
3. SRG analyzes the compressed key values and creates equation tables. (An *equation table* contains linear equations that relate the database keys with RAP numbers of the database.)
4. SRG creates a temporary SR load module in a work data set and simulates the invoking of the randomizing module by IMS. (The temporary SR is loaded into the extended private area.) This process is repeated until the optimization conditions (such as the number of synonyms) that you specify are satisfied.

   It then creates statistics reports on the usage of CIs and the distribution of synonyms.
5. SRG creates the SR source in SRRFILE. The SR source consists of a program part (SRR source) and a table part (SRT source) that contains equations. In DEDB, a table part is created for each DEDB area.
6. SRG assembles and link-edits the SR source to generate an SR load module in IMSVS.RESLIB or an authorized load module library that you specify.
   - SRG can create a DEDB SR as multiple modules. This function enables you to generate a DEDB SR whose total module size is more than 16 MB.
   - SRG can change the load module type of a DEDB SR from single type to multiple type or vice versa, once the DEDB SR load module of either type has been created.
   - You can select a reusability attribute for an HDAM SR and a PHDAM SR.

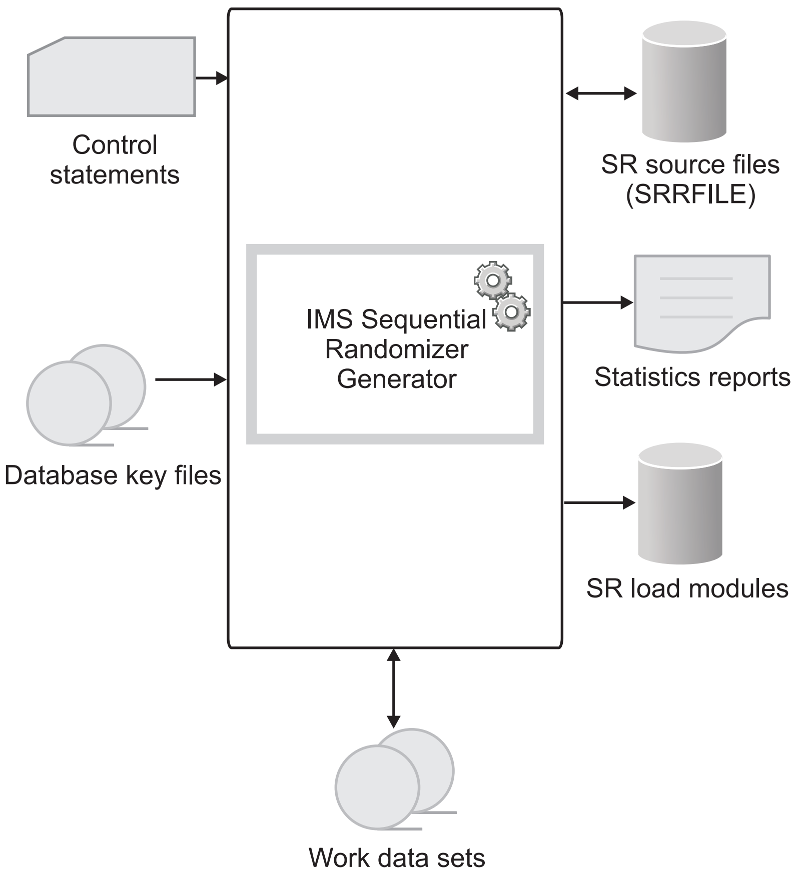


Figure 2. Overview of IMS Sequential Randomizer Generator

Database administration and change management solutions

IBM solutions help IT organizations maximize their investment in IMS databases while staying on top of some of today's toughest IT challenges. Database Administration and Change Management solutions can help maximize the management and use of your IMS databases.

Database Administration and Change Management are the core responsibilities of the DBA. If not managed correctly, they can monopolize data center resources, waste valuable time, and can result in the generation of unwanted errors.

In managing the database administration and change management process, database administrators are faced with many challenges like how do I:
- Ensure that I complete all of the necessary steps when making a change?
- Manage and track the changes to the definitions of my database objects?
- Propagate changes to other database environments?
- Keep IMS software versions current?
- Manage a corrupted database?
- Efficiently convert IMS full-function databases to the new High Availability Large Database (HALDB) format?

Many IMS Tools products can help reduce the negative impact data changes can have on your database. IMS Sequential Randomizer Generator is only one of several IMS Tools products that provide enhancements to the process of managing database operations.

Other IMS Tools products that can assist with database administration and change management include:
- IMS Configuration Manager
- IMS HALDB Toolkit
- IMS Online Reorganization Facility
- IMS Sysplex Manager
- Tools Base

For more information about the database administration and change management solutions for IMS, see the following web page:

http://www.ibm.com/software/data/db2imstools/solutions/database-admin.html

Program structure

The main program of SRG is FABOMAIN. SRG operates as a standard MVS™ batch job. This program is intended to run unauthorized with the exception that the randomizing module created by SRG will reside in an authorized library on IMS execution. The generated SR works in the same way as any other IMS randomizing modules.

Required storage areas

SRG requires the following virtual storage areas below 16 MB line:

Storage used by the module and areas acquired by the module	Approx. 30 KB (KB equals 1024 bytes)
Storage required for assembly and link edit	Approx. 120 KB
Storage required for the access method	Approx. 700 KB
Total	Approx. 850 KB

SRG requires the following virtual storage areas above 16 MB line:

- SRG creates a temporary SR and loads it into the extended private area (above 16 MB line) to simulate the IMS execution. Therefore, SRG requires additional storage to load the temporary SR. See Chapter 6, “Reference: Required amount of resources,” on page 107 to estimate the size of the temporary SR.
- If you specify that the compressed key file is to be created on storage, the storage for the data set is required in the extended private area. See KEYNBR for details.
- If you specify the AREADEF control statement, additional storage is required. See Chapter 6, “Reference: Required amount of resources,” on page 107 to estimate the size of work storage.

Restrictions

Certain restrictions apply to the use of IMS Sequential Randomizer Generator.

- SRG generates randomizing modules for a DEDB, an HDAM database, or a partition of a PHDAM database.
- SRG can generate a randomizer for DEDB that has up to 240 areas. DEDB that has more than 240 areas is not supported.
- The SR generated by SRG must reside in an authorized library on IMS execution.
- The SR generated by SRG refers to IMS control blocks to use ISWITCH and IMODULE functions provided by IMS, except when the SR is used by the following MVS batch utilities:
  - The following tools of IBM IMS Fast Path Solution Pack for z/OS® (program number 5655-W14):
    - The Unload function, the Reload function, or the Change function of IMS Fast Path Advanced Tool
    - The DEDB Unload utility, the DEDB Reload utility, or the DEDB Tuning Aid utility of IMS Fast Path Basic Tools
  - MSDB-to-DEDB Conversion utility of IMS
- The DEDB randomizer generated by SRG calculates the RAP number in two stages. In the first stage, it selects an area based on the database key value that was passed from IMS, then in the second stage, it calculates the RAP number within the selected area. In the second stage, the Sequential Randomizing Table (SRT) for the selected area is used for the calculation. The SRG DEDB randomizer, although it calculates RAP numbers in two stages, is not exactly the same as the randomizers that are referred to as two-stage randomizers in *IMS Database Administration*.

  A two-stage randomizer, in terms of IMS, allows a change to the root addressable definition for an individual area UOW during an area-level online change. Although an SRG DEDB randomizer determines the RAP number in two stages, it cannot use the new definition of RAP CIs even if you change the DBD and dynamically change the total number of RAPs for an area through an area level online change.

  However, if the area-level online change in your operational procedure will not change the total number of RAPs for an area, you can define an SRG DEDB randomizer as a two-stage randomizer during the DBDGEN phase. The SRT and the calculation for the RAP number are affected only by the total number of RAPs for the area, and the number of CIs that is defined for the dependent overflow (DOVF) section or for the independent overflow (IOVF) part will not affect the calculation. If you change the number of RAPs in an area, you must perform a database-level online change to change the old randomizer to the new randomizer that has been generated for the new area definition.

  See also Using SR under DEDB Online Change environment.

Processing environment

Verify that your hardware and software meet or exceed the minimum requirements for IMS Sequential Randomizer Generator.

Software requirements

SRG operates in a z/OS environment. The required release level of z/OS is the same as that for a version of IMS with which SRG is used.

SRG requires the following products and resources:

- One of the currently supported versions of IMS

  Note: In these topics, all supported versions of IMS are referred to as IMS, except where distinctions among them need to be made.
- One of the currently supported versions of DFSORT (5740-SM1) or a functionally equivalent sort/merge program
- One of the currently supported versions of High Level Assembler for z/OS, z/VM®, and z/VSE® (5696-234)
- One of the currently supported versions of High Level Assembler for z/OS, z/VM, and z/VSE Toolkit Feature (5696-234)
- The SR that is generated by SRG running under a currently supported version of IMS

The SR must be generated in the IMS environment in which the SR is used because the SR refers to the IMS control blocks.

Restriction: SRG supports the batch caller interface for the XCI DEDB randomizer only under IMS Version 10 or later.

Hardware requirements

The hardware requirements are the same as those for IMS.

Service updates and support information

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

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

http://www.ibm.com/support/entry/portal/Overview/Software/Information_Management/IMS_Tools

IMS Sequential Randomizer Generator 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

The IMS Tools Product Documentation web page provides current product documentation that you can view, print, and download. To locate publications with the most up-to-date information, refer to the following web page:

http://www.ibm.com/support/docview.wss?uid=swg27020942

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

http://www.redbooks.ibm.com

The Data Management Tools Solutions website shows how IBM solutions can help IT organizations maximize their investment in IMS databases while staying ahead of today's top data management challenges:

http://www.ibm.com/software/data/db2imstools/solutions/index.html

Receiving documentation updates automatically

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

To register with the My Notifications service:

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

How to send your comments

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

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

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.

The major accessibility features in IMS Sequential Randomizer Generator enable users to:

- Use assistive technologies such as screen readers and screen magnifier software. Consult the assistive technology documentation for specific information when using it to access z/OS interfaces.
- Customize display attributes such as color, contrast, and font size.
- Operate specific or equivalent features by using only the keyboard. Refer to the following publications for information about accessing ISPF interfaces:
  - *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.

Chapter 2. Operating instructions

The following topics describe how to run IMS Sequential Randomizer Generator to create a Sequential Randomizer (SR). These topics also describe the error conditions that might occur during the IMS Sequential Randomizer Generator run and during the execution of IMS using the generated SR.

Topics:

Preparation for running SRG

Before you run IMS Sequential Randomizer Generator, you must complete the following steps.

Procedure

1. Check the macro libraries.

   SRG uses the following four types of macro libraries. The data sets of the macro libraries are defined in the ddname MACLIB in the IBM-supplied cataloged procedure (FABOSRG). The macro libraries must be in the following sequence. If the block size of the SRG macro library is smaller than that of other macro libraries, specify the largest block size in the MACBLK parameter of FABOSRG.

   a. SRG macro library (HPS.SHPSMAC0 library)
   b. IMS macro library (SDFSMAC)
   c. High Level Assembler Toolkit macro library (SASMMAC2)
   d. System macro library (SYS1.MACLIB and SYS1.MODGEN libraries)

2. Check the sort library.

   SRG uses DFSORT (Data Facility Sort) when creating the equation tables; therefore, the sort library must be prepared in advance and the data set name must be specified in the VSSORT parameter of the FABOSRG cataloged procedure.

3. Allocate the randomizing module source library (SRRFILE).

   The data set for storing source code of the randomizing module (SRR/SRT source) must be allocated. The data set attributes are as follows:

   - Record format = FB
   - Block size = 4000 bytes (It can be changed by the SRRBLK parameter of the cataloged procedure FABOSRG.)
   - Record size = 80 bytes

   Important: Keep this data set for further use by IMS Sequential Randomizer Generator.

4. Generate the database key file that serves as input for SRG.

   The input database key file that is to be analyzed by SRG must be created by the user in advance. For the format of records in this data set, refer to "Database key file" on page 55.

5. Code the control statements, in which you specify the database key file, randomizing module to be generated, and the type of database. Control statements are specified in SYSIN. Refer to "Control statements" on page 28 for the details.
6. Code the JCL. An IBM-supplied cataloged procedure FABOSRG is provided.

What to do next

After all steps are performed, run the SRG job.

Attention: Read "Considerations before running SRG" on page 58 carefully before going on to the following steps. This topic contains some important information that should be considered in preparing for an SRG run.

Input and output summary

Use this topic to learn the main data sets and control statements used by SRG.

Input

Control statements
: Control statements specify the database key file, the randomizer to be generated, and the type of database. Control statements are specified in SYSIN.

Database key file
: This data set contains the database keys to be analyzed.

Reference

System macro library (SYS1.MACLIB and SYS1.MODGEN)
: These data sets contain the system macros. They are used during the randomizing module assembly.

SRG macro library (HPS.SHPSMAC0)
: This data set contains the SRG macros. It is used during the randomizing module assembly.

IMS macro library (SDFSMAC)
: This data set contains the IMS macros. These macros are used during the randomizing module assembly.

HLASM Toolkit macro library (SASMMAC2)
: This data set contains the HLASM Toolkit macros. These macros are used during the randomizing module assembly.

IMS load module library (SDFSRESL)
: This data set contains the IMS load modules. The IMS Callable Service module DFSCSI00 is linked with a randomizing module if it is generated as an XCI randomizing module.

DFSORT program library (SYS1.SORTLIB)
: This data set is used when sorting the work data set created while analyzing the database keys.

SRG load module library (HPS.SHPSLMD0)
: This data set contains the SRG load modules.

Output

Randomizing module source library (SRG.SRRFILE)
: This data set is used for storing the source code of the randomizing module (SR source). It is also used as an input to SRG when creating the SR load module. Keep a permanent copy of this data set.

Randomizing module load module library
: This data set is used for storing the final load module of the SR.

SYSOUT
: This data set contains the statistical data created as the result of randomizer simulation.

Job control language

To run SRG, supply an EXEC statement and the appropriate DD statements. JCL for running SRG is explained in this topic.

The following table summarizes the DD statements.

Table 1. SRG DD statements

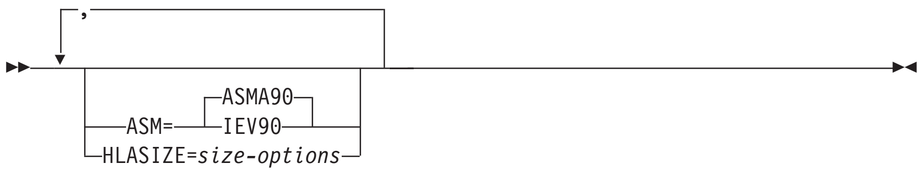
DDNAME	Use	Format	Need
STEPLIB	Input	PDS	Required
LKEDLIB	Input	PDS	Required
IN	Input	Specified in ANALYZE control statement	Optional
SYSIN	Input	LRECL=80	Required
MACLIB	Input	LRECL=80	Required
SORTLIB	Input	PDS	Required
SYSUT1 SYSUT2 SYSUT3	Work data set	-	Required
CARD	Work data set	LRECL=80	Required
COMPSR	Work data set	LRECL=80	Required
PARAFILE	Work data set	LRECL=80	Required
SOBJSET	Work data set	LRECL=80	Required
LKEDIN	Work data set	LRECL=80	Required
SYSLIN	Work data set	LRECL=80	Required
LKEDOUT	Work data set	-	Required
KEY	Work data set	-	Required
EQF	Work data set	LRECL=80	Required
OUTS	Work data set	-	Required
EQT	Work data set	-	Required
WORK	Work data set	-	Required
ASMIN	Work data set	LRECL=80	Required
SORTWK01 SORTWK02 SORTWK03 SORTWK04 SORTWK05	Work data sets	-	Required
SRRFILE	Input/output	LRECL=80	Required
IMSLIB	Input/output	PDS	Required
PRINT	Output	SYSOUT	Required
PRINT1	Output	SYSOUT	Required
SYSPRINT	Output	SYSOUT	Required
SYSUDUMP	Output	SYSOUT	Optional
SYSDUMY	Output	SYSOUT	Required
SYSPUNCH	Output	SYSOUT	Required

Note: In FABOSRG, all the data sets except the SYSIN data set are defined.

EXEC This statement must be in the following form:

```
//     EXEC PGM=FABOMAIN,PARM='parm_string'
```

The PARM='*parm_string*' is optional. If it is specified, the syntax must be as follows:



The ASM parameter specifies the name or an alias of the assembler program to be used. If the parameter is not specified, "ASMA90" is assumed.

The string '*size-options*' specified by the HLASIZE parameter is passed to the assembler program as arguments of the SIZE option when the program is invoked to assemble randomizer source files. For example,

- If `HLASIZE=(MAX,ABOVE)` is specified, the `SIZE` option will be `SIZE(MAX,ABOVE)`.
- If `HLASIZE=MAX-10M` is specified, the `SIZE` option will be `SIZE(MAX-10M)`.

STEPLIB DD
This required DD statement specifies the SRG load module library.

LKEDLIB DD
This required DD statement specifies the SRG load module library, which is generally the same as the library that is specified on the STEPLIB DD statement, and the IMS load module library (SDFSRESL).

IN DD
This optional input data set contains the input database keys that are sorted in ascending sequence. The format of this data set should be defined by the ANALYZE control statement parameters.

When each DEDB area has its own database key file, specify 'NULLFILE' for the DSN= parameter. Then define each database key file by specifying the DD statement with the area name as its ddname.

Note: This data set is not required when SRGEN=YES is specified on the ANALYZE control statement for an HDAM or PHDAM database, or when the AREADEF control statement is *not* specified (that is when only the ANALYZE control statement and LINKSR control statement are specified) for a DEDB database. Otherwise, this data set is required.

SYSIN DD
This required input data set contains SRG control statements.

MACLIB DD
Libraries of following three types must be defined:

- SRG macro library
- IMS macro library
- System macro library

These libraries are concatenated in the following order:

1. SRG macro library (HPS.SHPSMAC0)
2. IMS macro library (SDFSMAC)
3. HLASM Toolkit macro library (SASMMAC2)
4. System macro library (SYS1.MACLIB)
5. System macro library (SYS1.MODGEN)

BLKSIZE, if coded, must be the largest block size of the five data sets.

SORTLIB DD
: This required input partitioned data set (PDS) is the DFSORT (Data Facility Sort) program library. It is usually SYS1.SORTLIB.

SYSUT*n* DD
: These are required work data sets used for assembling and link-editing.

CARD DD
: This required work data set contains the final SR source. BLKSIZE, if coded, must be a multiple of 80.

COMPSR DD
: This is a required work data set. BLKSIZE, if coded, must be a multiple of 80.

PARAFILE DD
: This is a required work data set. BLKSIZE, if coded, must be a multiple of 80.

SOBJSET DD
: This is a required work data set for assembling and link-editing. BLKSIZE, if coded, must be a multiple of 80.

LKEDIN DD
: This is a required work data set. BLKSIZE, if coded, must be a multiple of 80.

SYSLIN DD
: This DD must specify SOBJSET and LKEDIN. SOBJSET must be specified preceding the LKEDIN.

LKEDOUT DD
: This is a required work data set.

KEY DD
: This required work data set contains the compressed keys that are created after the key analysis. You can create this data set either on DASD or on storage. To create the data set on storage, specify 'NULLFILE' for the CKEYF parameter of the cataloged procedure FABOSRG. To create the data set on DASD, specify '&&KEY' for the CKEYF parameter. No other specification is allowed. See Chapter 6, “Reference: Required amount of resources,” on page 107 to determine the size of this data set.

EQF DD
: This is a required work data set. BLKSIZE, if coded, must be a multiple of 80. See Chapter 6, “Reference: Required amount of resources,” on page 107 to determine the size of this data set.

OUTS DD
: This is a required work data set. See Chapter 6, “Reference: Required amount of resources,” on page 107 to determine the size of this data set.

EQT DD
This is a required work data set. See Chapter 6, "Reference: Required amount of resources," on page 107 to determine the size of this data set.

WORK DD
This is a required work data set. See Chapter 6, "Reference: Required amount of resources," on page 107 to determine the size of this data set.

ASMIN DD
This is a required work data set. BLKSIZE, if coded, must be a multiple of 80.

SORTWKnn DD
These are required work data sets used by DFSORT (Data Facility Sort).

See Chapter 6, "Reference: Required amount of resources," on page 107 to determine the size of these data sets.

SRRFILE DD
This required input/output data set is a source library in which the SR source is stored. Specification of concatenated data sets is *not* allowed for this library. BLKSIZE, if coded, must be a multiple of 80.

IMSLIB DD
This required input/output partitioned data set (PDS) is a load module library in which the SR load module is stored.

PRINT DD
This required output data set contains SRG messages issued by SRG.

PRINT1 DD
This required output data set contains statistics reports produced by SRG.

SYSPRINT DD
This required output data set contains SRG control statement assembly listing and SR assembly/link-edit listings.

SYSUDUMP DD
This defines output from a system abend dump routine. It is used only when a dump is required. Although optional, it is highly recommended that you include this data set.

SYSDUMY DD
This required output data set contains the intermediate SR assembly listing, SR link-edit listing, and sort output listing.

SYSPUNCH DD
This required output data set may contain the intermediate SR object module output. This DD is usually specified as DUMMY.

JCL procedure

To run SRG, use the IBM-supplied cataloged procedure FABOSRG, which is in the HPS.SHPSJCL0 library, or prepare a similar procedure of your own.

Chapter 3, “Examples,” on page 81 assumes that the IBM-supplied cataloged procedure is used.

The following table describes the parameters for FABOSRG.

Table 2. FABOSRG parameters

Parameter	Required or optional	Default	Description
INPTDSN	Required		Specifies the data set name of the input database key file, sorted in ascending key order; or specifies 'NULLFILE' when a database key file is created for each DEDB area. When 'NULLFILE' is specified, each database key file must be specified by a DD statement whose ddname is an area name. When a PHDAM SR is generated, the keys in the input key file must be only the keys in the target partition to which the generated SR is used.
SERIN	(Required)		Specifies the volume serial number of the input database key file. When 'NULLFILE' is specified for the INPTDSN parameter, this parameter is not required.
INUNIT	(Required)		Specifies the unit of the input database key file. When 'NULLFILE' is specified for the INPTDSN parameter, this parameter is not required. Example: INUNIT=SYSDA
NUM	Optional	1, SL	Specifies the sequence number and label type of the input database key file.
SMAC	Optional	SYS1.MACLIB	Specifies the data set names of the system macro library.
SMAC2	Optional	SYS1.MODGEN	
UMAC	Optional	HPS.SHPSMAC0	Specifies the data set name of the SRG macro library.
SPMAC	Optional	IMSVS.MACLIB	Specifies the data set name of the IMS macro library of your installation.
SPMAC2	Optional	IMSVS.GENLIBB	Specifies the data set name of the SASMMAC2 macro library of HLASM Toolkit.
IMSLIB	Optional	IMSVS.RESLIB	Specifies the data set name of the library in which the final SR load module is stored.
SOUT	Optional	A	Specifies the output class of SRG output defined in the ddnames PRINT, PRINT1, and SYSPRINT.
ACT	Optional	0,0	Specifies the SRG execution account number.
REGN	Optional	4096	Specifies the SRG executing region size in K bytes.
TME	Optional	1439	Specifies the SRG execution TIME parameter.
IP	Optional	10	Specifies the number of primary DASD cylinders for allocating the work data sets during SRG execution. ddname: KEY, OUTS, EQT, WORK

Table 2. FABOSRG parameters (continued)

Parameter	Required or optional	Default	Description
IU	Optional	SYSDA	Specifies the unit name for the work data sets. ddname: KEY, OUTS, EQT, WORK
IS	Optional	5	Specifies the number of secondary DASD cylinders for allocating the work data sets during SRG execution. If IP and IS values are too small, abend B37 will occur. In such a case, increase the values of these parameters. ddname: KEY, OUTS, EQT, WORK
VSSORT	Optional	SYS1.SORTLIB	Specifies the data set name of the load module library containing the DFSORT program library.
SDOLOAD	Optional	HPS.SHPSLMD0	Specifies the data set name of the SRG load module library.
PRMSRT	Optional	10	Specifies the number of primary DASD cylinders for allocating the sort work data set.
SECSRT	Optional	5	Specifies the number of secondary DASD cylinders for allocating the sort work data set. If PRMSRT and SECSRT values are too small, abend B37 will occur. In such a case, increase the values of these parameters.
SORTDV	Optional	SYSDA	Specifies the unit name for allocating the sort work data set.
SRSORC	Optional	SRG.SRRFILE	Specifies the name of the data set in which the SR source is stored. The attributes of this data set are as follows: RECFM=FB, LRECL=80 BLKSIZE value is specified by the SRRBLK parameter.
MACBLK	Optional	27920	Specifies the block size of the SRG macro library. If the block size of this library is smaller than that of other macro libraries, the largest block size should be specified.
SRRBLK	Optional	4000	Specifies the block size of the SR source library (ddname: SRRFILE).
EQFBLK	Optional	4000	Specifies the block size of the work data set (ddname: EQF).
ASMBLK	Optional	4000	Specifies the block size of the work data set (ddname: ASMIN).
CBLK	Optional	800	Specifies the block size of the work data set (ddname: CARD).
CP	Optional	10	Specifies the number of primary DASD tracks for allocating the work data set (ddname: CARD).
CS	Optional	5	Specifies the number of secondary DASD tracks for allocating the work data set (ddname: CARD).
JP	Optional	10	Specifies the number of primary DASD cylinders for allocating the work data sets (ddname: EQF, ASMIN).
JS	Optional	5	Specifies the number of secondary DASD cylinders for allocating the work data sets (ddname: EQF, ASMIN).

Table 2. FABOSRG parameters (continued)

Parameter	Required or optional	Default	Description
LSPC	Optional	800	Specifies the request type of the allocation of work data set (ddname: LKEDIN), which is one of the following: • TRK, • CYL, • block length.
LP	Optional	20	Specifies the number of the primary quantity for allocating the work data set (ddname: LKEDIN) as follows: • For LSPC=TRK, the number of tracks to be allocated. • For LSPC=CYL, the number of cylinders to be allocated. • For LSPC=block length, the number of data blocks in the data set.
LS	Optional	10	Specifies the number of additional tracks, cylinders or blocks to be allocated, if more space is needed for the work data set (ddname: LKEDIN).
LOSPC	Optional	TRK	Specifies the request type of the allocation of work data set (ddname: LKEDOUT), which is one of the following: • TRK, • CYL, • block length.
LOP	Optional	5	Specifies the number of the primary quantity for allocating the work data set (ddname: LKEDOUT) as follows: • For LOSPC=TRK, the number of tracks to be allocated. • For LOSPC=CYL, the number of cylinders to be allocated. • For LOSPC= block length, the number of data blocks in the data set.
LOS	Optional	5	Specifies the number of additional tracks, cylinders or blocks to be allocated, if more space is needed for the work data set (ddname: LKEDOUT).
LOD	Optional	5	Specifies the number of 256-bytes records needed in the directory of the work data set (ddname: LKEDOUT).
CKEYF	Optional	&&KEY	Specifies where to create the compressed key file as follows: **&&KEY** Specifies that the compressed key file is to be created on DASD. **'NULLFILE'** Specifies that the compressed key file is to be created in storage.
INDISP	Optional	'OLD,PASS'	Specifies the DISP parameter of the database key file. When multiple jobs refer to one database key file, 'SHR' should be specified (for DASD only).

Table 2. FABOSRG parameters (continued)

Parameter	Required or optional	Default	Description
ASM	Optional	ASMA90	Specifies the name of the assembler program. The supported names are ASMA90 and IEV90. This statement is provided for compatibility with the FABOSDO procedure of DBT Version 2 SDO.
HLASIZE	Optional	(MAX,ABOVE)	Specifies the string that is passed to the assembler program as arguments of the SIZE option when the assembler program is invoked to assemble randomizer source files.
IMSRES	Optional	IMSVS.RESLIB	Specifies the IMS load module library.

Note: To determine the values to be specified for IP, IS, PRMSRT, SECSRT, JP, JS, LSPC, LP, LS, LOSPC, LOP, LOS, and LOD, see Chapter 6, “Reference: Required amount of resources,” on page 107.

Input

The following topics describe all the input that you must specify to run SRG. It includes a control statement data set (SYSIN DD) and a database key file (IN DD).

- “Control statements”
- “Database key file” on page 55

Control statements

The SYSIN data set contains your specifications of a DEDB, an HDAM database, or a partition of a PHDAM database to be processed by SRG. It also contains user's options.

The SYSIN data set consists of the following control statements and parameters:

ANALYZE
The ANALYZE control statement is always required for generating SR for HDAM, PHDAM partition, or DEDB. The following information is specified:

- Characteristics for the input database key file

  Note: When you generate an SR for a PHDAM partition, you must specify the key file that contains the keys in that partition.
- Database types or data set organization

  Note: For an SR for HDAM or a PHDAM partition, the data set organization, either VSAM or OSAM, must be specified.
- Method of generating the equation table
- Printing option of output reports and listings
- Specifications for the equation table

AREADEF
The AREADEF control statement is required only when generating SR for DEDB. It provides the following information for each DEDB area:

- Range of keys for the area
- DEDB area types and formats
- Specifications for the equation table

LINKSR
The LINKSR control statement is required only when an SR is being generated for DEDB. It specifies the DEDB area names in the database.

Note: You can code the END statement of the assembler language to indicate the end of the SYSIN data set. The END statement is optional, but it is recommended that you code the END explicitly to mark the end of the SYSIN data set. See Chapter 3, “Examples,” on page 81 for some examples.

The ANALYZE control statement is required when you run SRG. Also, the TYPE parameter of the control statement is used to specify the database type that you are to generate. If the control statement is for an HDAM database or a partition of a PHDAM database, you can enter VSAM or OSAM as the keyword on the TYPE parameter. If the control statement is for DEDB, enter DEDB as the keyword on the TYPE parameter.

- HDAM or PHDAM (TYPE=VSAM or TYPE=OSAM)

No control statement other than the ANALYZE statement is required for HDAM and PHDAM.

The SRGEN parameter indicates how the SR should be generated.

SRGEN=NO (default)
SRG analyzes the input keys, creates an equation table, and generates the SR load module.

SRGEN=YES
SRG generates the SR load module using the SR source in SRRFILE, without analyzing the input keys.

- DEDB (TYPE=DEDB)

  There are two steps in SR generation for DEDB database. One is the *DEDB area processing*; SRG analyzes DEDB area keys and creates an equation table for each area. An SR source is created and stored in the SR source library (SRRFILE) at this time. The other step is the *SR load module generation*; SRG generates an SR load module using the SR source stored in the SR source library (SRRFILE).

  You can specify whether SRG performs these steps in one job or separately, by the combinations of the ANALYZE, AREADEF, and LINKSR control statements. In any case, the ANALYZE control statement is always required. See Table 3 on page 30.

DEDB area processing ((ANALYZE) + (AREADEFs))
SRG analyzes DEDB area keys for the specified areas and creates an equation table for each area. The SR source is created and stored in the SR source library. The SR load module is not generated.

Specify an ANALYZE control statement, and an AREADEF control statement for each DEDB area. By specifying multiple AREADEF control statements, multiple DEDB areas can be processed in a job. After all the DEDB areas have been processed, the user should generate an SR load module by *SR load module generation* processing.

AREADEF control statements must be specified *after* the ANALYZE control statement.

SR Load Module Generation ((ANALYZE) + (LINKSR))
SRG generates an SR load module using the SR source in SRRFILE, which is created during the *DEDB area processing*. Key analysis or equation table creation is not performed.

Specify an ANALYZE control statement and a LINKSR control statement. In this case, the DEDB area processing must have been completed for all the DEDB areas.

The LINKSR control statement must be specified *after* the ANALYZE control statement.

DEDB Area Processing and SR Load Module Generation (ANALYZE) + (AREADEF(s)) + (LINKSR)
SRG analyzes DEDB area keys, creates equation tables, and generates an SR load module in one job.

Specify an ANALYZE control statement, an AREADEF control statement for each DEDB area, and a LINKSR control statement. The LINKSR control statement must be specified *after* the AREADEF control statement.

Table 3. SRG Control Statement Combinations (DEDB)

DEDB area processing only	Generate SR load module only	DEDB area processing and SR load module generation in one job
ANALYZE AREADEF : : AREADEF	ANALYZE LINKSR	ANALYZE AREADEF : : AREADEF LINKSR

Control statement syntax

You must specify SRG control statements according to the assembler language coding conventions, which are outlined in this section.

Control statements must be coded between column 1 - 71. When a control statement continues on a next record, the continued record must be coded between column 16 - 71.

```
1-------10----16--------------------------------------------------------------72

label   ANALYZE  (1,4,U,F0F1F2F3),                                             x
               INKEYL=4,                                                       x
               INLRECL=80
```

- To continue a control statement, enter a nonblank character in column 72, and continue the statement from column 16 on the next line. Columns to the left of the continuation column must be blank. If the statement continues to more than two lines, column 72 of each line except for the last must contain a nonblank character.
- A comment statement must start with an asterisk in column 1.
- A comment must be placed between column 1 - 71.
- A comment can be placed on the same line as a control statement by separating each other with one or more blanks.

Notational conventions

The following symbols should be coded as they appear in the control statement format:

comma
  ,

equal sign
  =

parentheses
  ( )

The following symbols are used to define the control statement format:

brackets [ ]
  Indicates an optional parameter.

ellipsis ...
: Indicates that the preceding item may be repeated more than once.

ANALYZE control statement

The ANALYZE control statement is always required for generating an SR, whether for DEDB databases or for HDAM databases, including PHDAM databases.

The syntax of the ANALYZE control statement for HDAM (including PHDAM) is as follows:

```
label ANALYZE (a,b,c[,d]),(a,b,c[,d]),...
             ,TYPE=[OSAM|VSAM]
             ,INKEYL=
             ,INLRECL=
             ,INBLKSZ=
             ,INOFFST=
             [,DBLOFFST=]
             [,CIBYTE=]
             [,CITRK=]
             [,TRCYL=]
             [,PRIMCI=]
             [,FRECI=]
             [,DBRECSZ=]
             [,OPTN=]
             [,OPTMZ=]
             [,AREANM=]
             [,KEYNBR=]
             ,CORE=
             [,TLIMIT=]
             [,TNUMLIM=]
             [,OPTMTIME=]
             [,SIZECHK=]
             [,SRGEN=]
             [,ERPROC=]
             [,PRINTNG=]
             [,RMODE=]
             [,LNKPGM=]
             [,ATTRIB=]
```

The syntax of the ANALYZE control statement for DEDB is as follows:

```
label ANALYZE (a,b,c[,d]),(a,b,c[,d]),...
             ,TYPE=DEDB
             ,INKEYL=
             ,INLRECL=
             ,INBLKSZ=
             ,INOFFST=
             [,DBLEN=]
             [,NOAREA=]
             [,OPTN=]
             [,OPTMZ=]
             [,AREANM=]
             [,KEYNBR=]
             [,DBM=]
             [,SIZECHK=]
             [,ERPROC=]
             [,PRINTNG=]
             [,RMODE=]
             [,SRCMIG=]
             [,SRTYPE=]
             [,TYPECHG=]
             [,LNKPGM=]
             [,CALLTYPE=]
```

The following sections describe each parameter.

- "Name and type of randomizing module"
- "For input database key file specifications"
- "For database characteristics specification" on page 39
- "For SR and equation table generation method specifications" on page 40
- "For output report/listing specifications" on page 45
- "For compatibility with IAPP SDO" on page 45
- "For SR load module type specifications" on page 46
- "For linker program specifications" on page 47
- "For randomizer call interface specifications" on page 47

Name and type of randomizing module

label **(HDAM/PHDAM/DEDB)**

Starts from column 1 and specifies the name of the randomizing module (SR) to be created. This name can contain up to eight alphanumeric characters, of which the first is a letter. It is also specified in the RMNAME= parameter of the DBD statement of DBDGEN.

This name cannot be the same as the label on the AREADEF control statement.

TYPE **(HDAM/PHDAM/DEDB)**

Specifies the type of the database from one of DEDB, OSAM, or, VSAM. For an HDAM or PHDAM database, the access method used for the database or the partition—either OSAM or VSAM—must be specified.

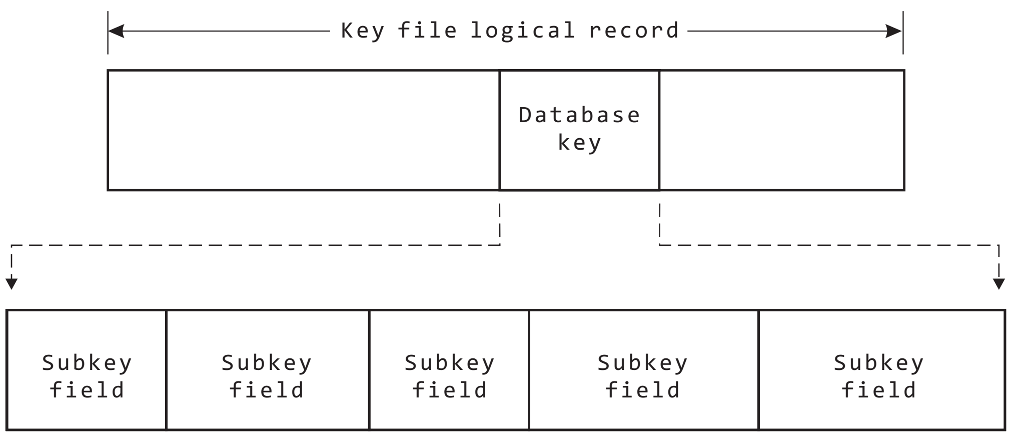
Default: DEDB

For input database key file specifications

For the following parameters that require numeric values, specify them in decimal numbers, unless otherwise stated.

(a,b,c) or (a,b,c,d) **(HDAM/PHDAM/DEDB)**

This is a required positional parameter used for database key compression. It specifies the attributes of subkey fields in the input database key file. A database key field consists of 1 to 25 subkey fields. For each subkey field, a start column, end column, and attribute are specified.



a Specifies the start column of the subkey field within the database key field. This value must be between 1 and 255. If there is more than one subkey field, the value for the first subkey field must be 1.

b Specifies the end column of the subkey field within the database key. This value must be between the value specified in **a** and 255.

c Specifies the attribute of the subkey field. The attribute is specified with a symbol described below.

d Used only when the attribute U is specified.

Note that **a** and **b** are not offsets within a logical record in the database key file; they are positions (starting from 1) within the database key field.

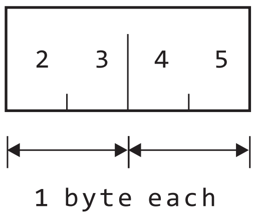
Subkey Field Attribute Parameter

X: Hexadecimal key field
Specified when no other attributes apply to the subkey field.

Maximum length: 254 bytes

P: Unsigned packed decimal key field
Specified when the subkey field consists of unsigned packed decimal data.

Example

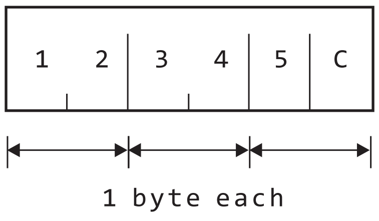


The upper and lower four bits of each byte of the specified subkey field must be in the range of B'0000' to B'1001'.

Maximum length: 4 bytes

S: Signed packed decimal key field
Specified when the subkey filed consists of signed packed decimal data; however, all data is treated as positive numbers.

Example



The upper and lower four bits, except the sign bits, of each byte of the specified subkey field must be in the range of B'0000' to B'1001'. The sign bits must be either B'1010', B'1100', or B'1111'.

Maximum length: 5 bytes

C: Alphanumeric key field

Specified when the subkey field consists of alphanumeric data (A to Z, 0 to 9, and a blank).

Maximum length: 255 bytes

N: Numeric key field

Specified when the subkey field consists of numeric data (0 to 9).

Maximum length: 9 bytes

I: Ignored key field

Specified when the subkey field should not be analyzed by SRG. Even if this attribute is specified, the input database key must be in ascending sequence and at least one of the other attributes must be specified.

When the generated SR analyzes input database keys, it ignores the subkey field with this attribute.

Maximum length: 254 bytes

A: Alphabetic key field

Specified when the subkey field consists of alphabetic data (A to Z) and a blank.

Maximum length: 255 bytes

Q: Character key field

Specified when the subkey field consists of special characters, alphanumeric characters (A to Z and 0 to 9), and a blank. Characters are:

A to Z, 0 to 9, blank, ¢ · ¬ < (+ | & ! $ *) : - / , % _ > ? : # @ ' = " }

See the following hexadecimal code matrix for the characters of this field:

```
           *** TYPE : Q ***

             0 1 2 3 4 5 6 7 8 9 A B C D E F   <-- bits 4567

         0   - - - - - - - - - - - - - - - -
         1   - - - - - - - - - - - - - - - -
bits     2   - - - - - - - - - - - - - - - -
0123     3   - - - - - - - - - - - - - - - -
         4   X - - - - - - - - - X X X X X X
  |----> 5   X - - - - - - - - - X X X X X X
         6   X X - - - - - - - - - X X X X X
         7   - - - - - - - - - - X X X X X X
         8   - - - - - - - - - - - - - - - -
         9   - - - - - - - - - - - - - - - -
         A   - - - - - - - - - - - - - - - -
         B   - - - - - - - - - - - - - - - -
         C   - X X X X X X X X X - - - - - -
         D   X X X X X X X X X X - - - - - -
         E   - - X X X X X X X X - - - - - -
         F   X X X X X X X X X X - - - - - -
```

Maximum length: 255 bytes

Y: Katakana key field

Specified when the subkey field consists of katakana (Japanese phonetic symbols) and a blank. See the following hexadecimal code matrix for the characters of this field:

```
             *** TYPE : Y ***

             0 1 2 3 4 5 6 7 8 9 A B C D E F   <-- bits 4567

         0   - - - - - - - - - - - - - - - -
         1   - - - - - - - - - - - - - - - -
bits     2   - - - - - - - - - - - - - - - -
0123     3   - - - - - - - - - - - - - - - -
         4   X X - - - - X X X X - X - - - -
  |----> 5   - X X X X X X - - - - - - - - -
         6   X - - - - - - - - - - X - - - -
         7   - - - - - - - - - - - - - - - -
         8   - X X X X X X X X X X - X X X X
         9   X X X X X X X X X X X - - X X X
         A   - - X X X X X X X X X - X X X X
         B   - - - - - - - - - - X X X X X X
         C   - - - - - - - - - - - - - - - -
         D   - - - - - - - - - - - - - - - -
         E   - - - - - - - - - - - - - - - -
         F   - - - - - - - - - - - - - - - -
```

Maximum length: 255 bytes

Z: Katakana and character key field

Specified when the subkey field consists of alphanumeric characters (A to Z, 0 to 9) and the characters contained in the attribute Y field.

Maximum length: 255 bytes

U: Specific character key field

Specified when the subkey field consists of specific characters. If this attribute is specified, the specific characters must be specified (in hex) in the fourth parameter, within the following range:

Maximum length: 123 bytes

Minimum length: 2 bytes

Example: Specify as follows when each byte of the subkey field is a number between 0 and 3 and the field length is 4 bytes:

(1,4,U,F0F1F2F3)

O: Nonsequential randomizing module

All of the previous parameters are for generating a *sequential* randomizing module. However, if attribute O is specified for the entire key field, a *nonsequential* randomizing module is created. If this attribute is specified, there is no relationship between the key sequence and relative RAP numbers. If attribute O is specified for one of the fields, the entire key field is processed as the field of O.

Specify as follows when the database key consists of the subkey fields as shown in Figure 3:

(1,4,A),(5,7,N),(8,10,U,F0F1),(11,13,Y),(14,17,Z)

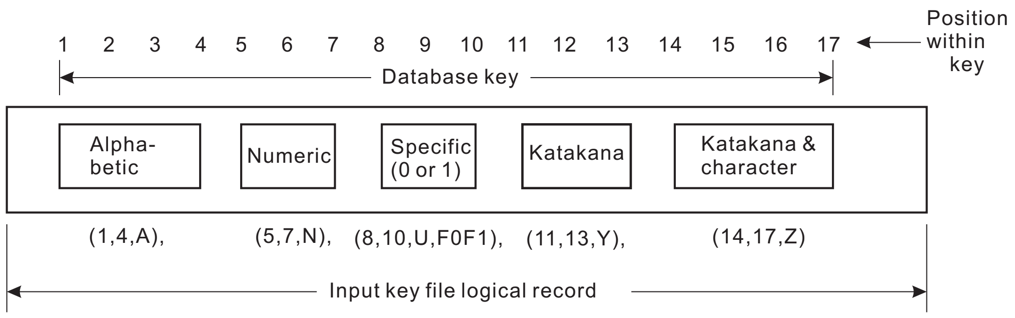


Figure 3. Key compression parameter specification example

INKEYL (HDAM/PHDAM/DEDB)

Specifies the length (in bytes) of the key field within the input database key file. This value must be the total length of all the subkey fields specified with the subkey field attribute parameter. This is a required parameter.

Maximum length: 255 bytes

INLRECL (HDAM/PHDAM/DEDB)

Specifies the length (in bytes) of the input database key file record. INLRECL must be equal to or greater than INKEYL. This is a required parameter.

INBLKSZ **(HDAM/PHDAM/DEDB)**

Specifies the block size (in bytes) of the input database key file. This value must be an integral multiple of the record length. This is a required parameter.

INOFFST **(HDAM/PHDAM/DEDB)**

Specifies the start column of the key field within the input database key file record. This is a required parameter. The following conditions must be satisfied:

```
INOFFST is equal to or greater than 1
(INOFFST + INKEYL - 1) is equal to or less than INLRECL
```

Example: If the block size is 800 bytes, record length is 80 bytes, key field start column is column 2, and the key length is 3 bytes, the specification is as follows:

```
INLRECL=80, INBLKSZ=800, INKEYL=3, INOFFST=2
```

DBLEN **(DEDB)**

Specifies the position of the field containing the database record length (see note) within the input database key record.

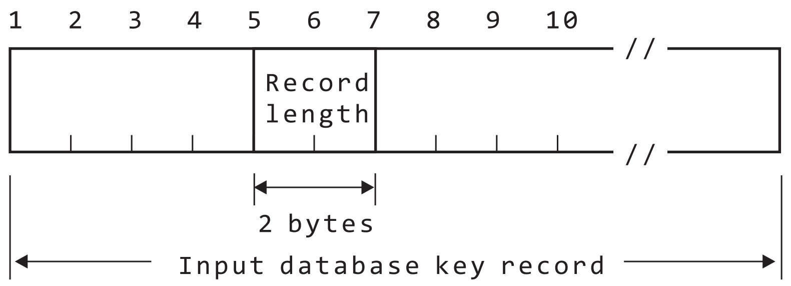
If specified, DBLEN must be 1 or greater and the field must be within the input database key record and specified so as not to overlap the key field. In addition, the database record length field must be 2 bytes, and its value must be equal to or greater than the database key length, and equal to or less than 65535 bytes.

If the DBLEN parameter is not specified, the AVRECL parameter of the AREADEF control statement must be specified.

If the DBLEN parameter of the ANALYZE statement and the AVRECL parameter of the AREADEF control statement are both specified, AVRECL is ignored.

Note: The database record length must include the length of the prefix of each segment.

Example



DBLEN=5

DBLOFFST **(HDAM/PHDAM)**

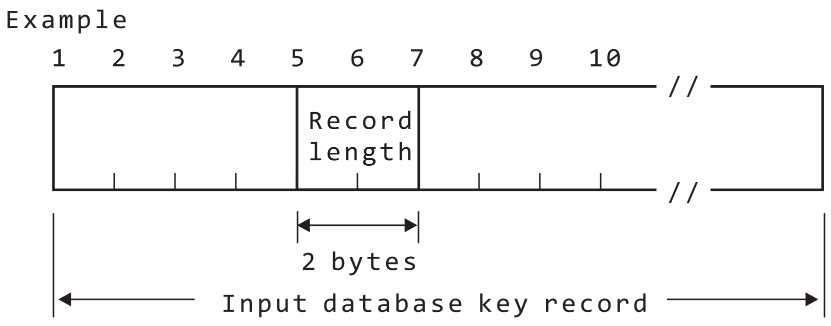
Specifies the position of the field containing the database record length (see note) within the input database key record.

If specified, DBLOFFST must be 1 or greater and the field must be within the input database key record and specified so as not to overlap the key field. In addition, the database record length field must be 2 bytes, and its value must be equal to or greater than the database key length, and equal to or less than 65535 bytes.

If the DBLOFFST parameter is not specified, the DBRECSZ parameter of the ANALYZE control statement must be specified.

If the DBLOFFST parameter and the DBRECSZ parameter are both specified of the ANALYZE control statement, DBRECSZ is ignored.

Notes:
- The database record length must include the length of the prefix of each segment.
- The DBLOFFST parameter supersedes the DBLEN parameter for HDAM and PHDAM SR after applying APAR PQ83931. The DBLEN parameter is supported only for the compatibility with the existing JCL streams.



```
DBLOFFST=5
```

For database characteristics specification

For the following parameters that require numeric values, specify them in decimal numbers, unless otherwise stated.

CIBYTE (HDAM/PHDAM)
Specifies the CI or block size (in bytes) of the root addressable area to be created. This value must be greater than the database key length.

Default: 4096

CITRK (HDAM/PHDAM)
Specifies the number of CIs or blocks per track of the root addressable area to be created. This value must be 1 or greater.

Default: 3

TRCYL (HDAM/PHDAM)
Specifies the number of tracks per cylinder of the root addressable area to be created. This value must be 1 or greater.

Default: 19

PRIMCI (HDAM/PHDAM)
Specifies the size (in cylinders) of the root addressable area to be created. It must be a number between 1 and 32767.

Default: 20

Note: In the case of HDAM, the number of CIs that can be used by the SR is calculated by the following formula:

PRIMCI × ((CITRK × TRCYL) - FRECI)

See Figure 26 on page 101 for the calculation of the number of RAPs.

FRECI **(HDAM/PHDAM)**
Specifies the number of CIs or blocks per cylinder that are not to be used by the SR load module. The value must satisfy the following condition:

$0 \leq \text{FRECI} < \text{CITRK} \times \text{TRCYL}$

This parameter is used to intentionally leave space within the database.

Default: 1

DBRECSZ **(HDAM/PHDAM)**
Specifies the average record length (in bytes) of the database to be stored in the root addressable area. This parameter must be specified when a value is not specified in the DBLOFFST parameter.

This value must be greater than the database key length, and must not be more than 65535. If both the DBRECSZ and DBLOFFST parameters are specified, the DBRECSZ parameter is ignored.

The average length must include the prefix of each segment.

NOAREA **(DEDB)**
Specifies the number of DEDB areas. The value must be a number in the range of 1 - 240.

If the NOAREA parameter is specified and an SR load module is to be created (LINKSR control statement is specified), the number of areas is checked. If the number of areas specified by the NOAREA parameter and the number of areas specified in the LINKSR control statement do not match, it causes a control statement error. If the NOAREA parameter is not specified, the number of areas is not checked.

Note: SRG can generate a randomizer for DEDB that has up to 240 areas. DEDB that has more than 240 areas is not supported.

For SR and equation table generation method specifications

For the following parameters that require numeric values, specify them in decimal numbers, unless otherwise stated:

OPTN **(HDAM/PHDAM/DEDB)**
Specifies the method of creating the equation table used by SR to calculate the RAP numbers for database keys.

The following three methods (Option 0, 1, or 3) are available:

0 This method is suitable for the case when the intervals between database keys are relatively uniform.

1 This method is suitable for the case when database keys are unevenly distributed.

3 This method is suitable for the case when database keys are unevenly distributed and SR is to be created in a short time.

For more detailed information, see Chapter 7, “Reference: Database key analysis methods,” on page 111.

Default: 3

OPTMZ (HDAM/PHDAM/DEDB)

OPTMZ=(a,b)

Specifies the maximum number of synonyms and the maximum CI capacity ratio used to perform optimization by synonym or CI capacity ratio.

a
Specifies the number of synonyms allowed per RAP. ($0 \le a \le 9999$)

Default: 9999

b
Specifies the maximum CI capacity ratio (%). ($1 \le b \le 100$)

Default: 100 (100% indicates that CI capacity ratio is not used for SR optimization)

Optimization is not performed if the OPTMZ parameter is omitted or if the default value is specified.

If the database type is DEDB and conflicting OPTMZ parameters are specified on the ANALYZE and AREADEF control statements, the value specified on the AREADEF control statement takes effect.

If the OPTMZ parameter is specified on the ANALYZE control statement, but omitted on the AREADEF control statement, the value specified on the ANALYZE control statement takes effect. The OPTMZ parameter is specified as follows:

- Specifies both the maximum number of synonyms and maximum CI capacity ratio:

  ```
  OPTMZ=(a,b)
  ```
- Specifies the maximum number of synonyms only:

  ```
  OPTMZ=(a)
  OPTMZ=(a,)
  OPTMZ=(a,100)
  OPTMZ=a
  ```
- Specifies the maximum CI capacity ratio only:

  ```
  OPTMZ=(,b)
  OPTMZ=(9999,b)
  ```

See “Optimizing the equation table” on page 100 for details on the maximum number of synonyms and maximum CI capacity ratio.

AREANM (HDAM/PHDAM/DEDB)

Specifies the name of the generated SRT with up to eight alphanumeric characters. This name is used as a CSECT name. This name is used as the name of the extended master table, if SRTYPE=MULTI is specified. The name must not be the same as the label on the ANALYZE or AREADEF statement. Furthermore, the name must not be equal to the name:

```
xxxxx##@
```

where xxxxx is the first five characters of the SR name specified on the label, padded with '#' on the right side if the SR name is four characters or less.

This name must be a unique name in the ECSA of the user's system.

See “Preparation for using SR under IMS” on page 71 for the details of how this name is used by the DEDB SR generated with SRTYPE=MULTI specified.

Default: xxxxx##T where xxxxx is the first five characters of the SR name specified on the label, or the SR name if the SR name is five characters or less.

Example:

1. When the SR name consists of six or more characters:
   SR name: SRRDEDB
   AREANM: SRRDE##T
2. When the SR name consists of five characters:
   SR name: SRRD1
   AREANM: SRRD1##T
3. When the SR name consists of four or less characters:
   SR name: SRR
   AREANM: SRR##T

KEYNBR (HDAM/PHDAM/DEDB)

Specifies that the compressed key file is to be created on storage. The compressed key file is a work data set which is used by SRG for analyzing database keys. For HDAM, specify the number of keys contained in the database key file. For DEDB, specify the number of keys in a DEDB area that contains the largest number of keys among all the areas. Value can be from 1 to 99999999.

When the KEYNBR parameter is specified, the compressed key file is created on private storage above 16 MB line. In this case, specify CKEYF = 'NULLFILE' in the JCL procedure FABOSRG so that the DASD space can be saved. When the KEYNBR parameter is not specified, the compressed key file is created on DASD (ddname: KEY), with the maximum block size available. Whether the compressed key file should be created on storage or on DASD depends on the SRG environment.

Default: Key file is created on DASD.

CORE (HDAM/PHDAM)

Specifies the size (in bytes) of the equation table. This is a required parameter for an HDAM database.

The number of equations should be in the range of 1 to 32767. The size of the equation table can be calculated as follows:

$$\text{CORE} = (\text{Number of equations}) \times (\text{Compressed key length} + 9)$$

The CORE value should be in the following range:

$$(\text{Compressed key length}) + 9 \leq \text{CORE} \leq (\text{Compressed key length} + 9) \times 32767$$

See Chapter 6, “Reference: Required amount of resources,” on page 107 to determine the CORE value.

TLIMIT (HDAM/PHDAM)

Specifies the maximum value of an SRT for the equation table optimization. The value should be from the INKEYL parameter value to 9999999. When the TLIMIT parameter is specified, and when the size of the SRT becomes equal to or greater than the maximum value during optimization, then SRG stops the optimization process and generates SR. In this case, the SR may not satisfy the optimization condition. The TLIMIT parameter should be specified on the ANALYZE control statement (HDAM/PHDAM) or on the AREADEF control statement (DEDB) only when the OPTMZ parameter is specified.

Default: 9999999

Note: When TLIMIT=9999999 is specified, the SRT is optimized to contain 32767 or fewer linear equations. If you want to further optimize the equation table when the number of equations in an SRT exceeds 32767, use the TNUMLIM parameter instead of the TLIMIT parameter.

TNUMLIM (HDAM/PHDAM)

Specifies the maximum number of equation tables of an SRT for equation table optimization. The TNUMLIM parameter cannot be specified with the TLIMIT parameter.

The TNUMLIM value must be in the following range:

0 ≤ TNUMLIM ≤ 16777215 / (Compressed key length + 9)

If the TNUMLIM parameter is specified, SRG stops the optimization processing when the number of equation tables of the SRT reaches the maximum value and generates an SR. In this case, the SR might not satisfy the optimization condition.

Specify the TNUMLIM parameter on the ANALYZE control statement (for HDAM or PHDAM) or on the AREADEF control statement (for DEDB) only when you specify the OPTMZ parameter.

When the TNUMLIM parameter is not specified or when TNUMLIM=0 is specified, the value of the TLIMIT parameter is used.

Default: 0

OPTMTIME (HDAM/PHDAM)

Specifies the maximum elapsed time in minutes for equation table optimization. Specify a value in the range of 0 - 1440.

If the OPTMTIME parameter is specified, SRG stops the optimization processing when the maximum time is reached and generates an SR. In this case, the SR might not satisfy the optimization condition.

Specify the OPTMTIME parameter on the ANALYZE control statement (for HDAM or PHDAM) or on the AREADEF control statement (for DEDB) only when you specify the OPTMZ parameter.

When the OPTMTIME parameter is not specified or when OPTMTIME=0 is specified, equation table optimization is processed without a time limit.

Default: 0

SIZECHK (HDAM/PHDAM/DEDB)

Specifies whether SRG issues a message and stops processing when SRG determines that the size of one of the modules that composes the SR will exceed 16 MB.

YES SRG issues the FABO0133A or FABO0134A message and ends with RC=8 when the size of the SR module exceeds 16 MB.

NO SRG continues processing even when the size of the SR module exceeds 16 MB. As a result, the assembler program issues the ASMA039S message and SRG ends processing with RC=8 after issuing the FABO0116A message.

Default: NO

Note: For details of the modules that compose an SR, see "Generating an SR load module" on page 105.

SRGEN (HDAM/PHDAM)
Specifies whether the SR is to be generated from scratch, or the existing SR source is used. In other words, SRG performs either the entire process of generating an SR (including the database key analysis), or only the assembling and link-editing of the SR source.

YES SRG uses the SR source library as an input to assemble and link-edit the existing SR source to generate an SR load module. Database key analysis or the equation table creation is not performed.

NO SRG generates SR by performing the entire process (database key analysis, equation table creation, assembling and link-editing of the SR source).

Default: NO

DBM (DEDB)

This parameter is provided only for the compatibility with prior products, and only DBM=DBT is accepted.

ERPROC (HDAM/PHDAM/DEDB)
Specifies whether the status code 'FM' is used.

CONT
If an unexpected database key is passed to the SR, SR returns X'04' as a return code to IMS, and the dependent region does not end abnormally. As a result, the application program can receive 'FM' as a status code. If the SR is passed a database key whose length does not match the one specified during the SR generation, then, regardless of this parameter specification, the SR always returns the return code X'08' to IMS and the dependent region abends.

ABEND
If an unexpected database key is passed to the SR, SR returns X'08' as a return code to IMS. Then, in the case of HDAM or PHDAM, the dependent region ends abnormally with the abend code U812; in the DEDB case, the dependent region ends abnormally with the abend code U1021.

See "SR action for an unexpected key" on page 77 for the types of unexpected keys.

In the case of DEDB, this parameter may be changed in the generation step of an SR load module where only ANALYZE and LINKSR control statements are specified. In the case of HDAM or PHDAM, this parameter may be changed in the SRG run with SRGEN=YES specified.

Default: ABEND

See "How the Sequential Randomizer (SR) works" on page 73.

For output report/listing specifications

PRINTNG (HDAM/PHDAM/DEDB)

The PRINTNG parameter selects the output reports/listings to be produced by SRG. Either YES, NO, or SHORT can be specified. See the following table for the list of output reports and listings produced by each option.

Default: YES

Table 4. PRINTNG parameter

Output report/listing	Out to (DDNAME)	PRINTNG=		
		YES	NO	SHORT
CI Distribution List report	PRINT1	O	X	X
Randomizer Statistics report	PRINT1	O	O	O
Synonym Distribution Status report	PRINT1	O	O	O
Recommended DBD Control Statement report	PRINT1	O	O	O
DEDB Master Table report	PRINT1	O	O	O
SRG Control Statement Assembly listing	SYSPRINT	O	O	O
SR Assembly listing • SRR part • SRT part • DBDGEN control statements	SYSPRINT	O	O (*)	X
SR Link-edit listing	SYSPRINT	O	O	O

Notes:

- O indicates that the report/listing is to be produced. X indicates that the report/listing is not to be produced.
- DEDB Master Table report is produced only for DEDB databases.
- (*): When PRINTNG=NO is specified, SR Assembly listing (SRR part) is produced PRINT GEN (macro expansion), and SR Assembly listing (SRT part) is produced PRINT NOGEN (no macro expansion).

For compatibility with IAPP SDO

RMODE (HDAM/PHDAM/DEDB)

This parameter is left to provide compatibility with the IMS/VS Aid Program Package (IAPP) SDO and is meaningless internally.

SRCMIG (DEDB)

Specifies that the SRRFILE should be migrated to a new structure and the SR is to be regenerated.

YES The current SRRFILE is to be migrated into a new structure and the SR is to be regenerated. The areas to be migrated are specified in the AREAID parameter of LINKSR control statement. Therefore, all the areas in SRRFILE must be specified in the AREAID parameter. When SRCMIG=YES, you cannot specify the LINKID parameter of LINKSR control statement or the AREADEF control statement.

NO Migration is not performed.

For more information, see Chapter 8, "Reference: Migration procedures," on page 123.

Default: NO

For SR load module type specifications

SRTYPE **(DEDB)**

Specifies whether the SR is to be generated as a single module, or as multiple modules.

SINGLE
SR load module is generated as a single module.

MULTI
SR load module is generated as multiple modules. In this case, the load modules are composed of the SRR module, the master table module, and the area equation table modules. One area equation table module is generated for each area.

The SR should be generated as a single module, except when the SR module size might exceed 16 MB, which is the maximum size of the load module to be generated by the linkage-editor. See "Generating an SR load module" on page 105 for the structure of the generated SR. See "Considerations before running SRG" on page 58 concerning when SRTYPE=MULTI should be specified. See also "Preparation for using SR under IMS" on page 71 for the details of how a DEDB SR generated with SRTYPE=MULTI specified is loaded by IMS.

Default: SINGLE

TYPECHG **(DEDB)**

Specifies whether or not to change the SR load module type. This parameter should be used with the SRTYPE parameter.

YES The load module type of a DEDB SR is changed to the type that is specified on the SRTYPE parameter. To perform the type change function, all source files and all load modules for the SR must exist in the specified libraries. When TYPECHG=YES, you cannot specify the LINKID parameter of LINKSR statement or the AREADEF control statement.

NO Type change is not performed.

See "Considerations before running SRG" on page 58 for the considerations when TYPECHG=YES is specified.

Default: NO

ATTRIB **(HDAM/PHDAM)**

Specifies the attribute of the SR. This parameter is valid only for HDAM and PHDAM SRs.

blank The SR is generated with the nonreusable attribute.

REUS The SR is generated with the reusable attribute.

RENT The SR is generated with the reentrant attribute.

For the details of the attribute of HDAM and PHDAM SRs, see "How the SR works for an HDAM or PHDAM database" on page 76.

Default: blank

For linker program specifications

LNKPGM **(HDAM/PHDAM/DEDB)**

Specifies whether COMPAT(LKED) option should be specified or not when DFSMS Binder is called internally to link-edit the SR.

LKED This option is accepted for the compatibility with prior products. This option has the same effect as BINDER option.

BINDER
Call DFSMS Binder without COMPAT(LKED) option.

Note: If this option is specified and the option LINKID=ALL is specified on the LINKSR control statement, and if you are to regenerate a DEDB SR that has been generated by the SDO of the level prior to APAR PN82345 for DBT Version 2 Release 2 SDO, you must delete the old modules of the SR from your SR module library (DDNAME: IMSLIB) before you run SRG.

BINDERLKED
Call DFSMS Binder with COMPAT(LKED) option.

Note: You must specify this option if you want to reassemble and relink, using the LINKSR statement, a DEDB SR that has been generated by the SDO of the level prior to APAR PN82345 for DBT Version 2 Release 2 SDO and if you do not reassemble all the equation tables for all areas — that is, if you do not specify all areas on the LINKID parameter of the LINKSR control statement. For an HDAM SR, this option is not required even if you want to generate the SR module from the existing source files with the SRGEN=YES option.

Default: LKED

Note: There is no difference between specifying LNKPGM=LKED and specifying LNKPGM=BINDER. In both cases, the linker program named "IEWL" is called without the COMPAT(LKED) option. Thus, you do not have to specify this parameter unless you need the COMPAT(LKED) option for the Binder.

For randomizer call interface specifications

CALLTYPE **(DEDB)**

Specifies the type of randomizer call interface. This parameter is valid only for DEDB randomizers.

STD The standard call interface is used. This option can be specified for the SR generated either as a single module (SRTYPE=SINGLE) or as multiple modules (SRTYPE=MULTI).

XCI Specifies that the Extended Call Interface is used when calls to the SR are made. You can specify this option only when you generate the SR as multiple modules (SRTYPE=MULTI). You must specify this option if you want to use the DEDB online change function for your SR of SRTYPE=MULTI. For information about specifying an XCI randomizer in DBDGEN, see *IMS System Utilities*.

Default: STD

AREADEF control statement

The AREADEF control statement is required only for DEDB SR generation. This control statement is required for each DEDB area. Up to 240 AREADEF control statements can be specified in a single SRG run.

Note: SRG can generate a randomizer for DEDB that has up to 240 areas. DEDB that has more than 240 areas is not supported.

The syntax of AREADEF control statement is shown as follows:

```
label   AREADEF  UOW=
                ,SIZE=
                ,ROOT=
                ,TSIZE=
                ,KEYRNG=|,ARETRY=
                [,AVRECL=]
                [,DELCHAR=]
                [,CVCHAR=]
                [,AREXT=]
                [,OPTMZ=]
                [,TLIMIT=]
                [,TNUMLIM=]
                [,OPTMTIME=]
```

Each parameter is described in this section. For the following parameters that require numeric values, specify them in decimal numbers, unless otherwise stated.

label **(DEDB)**

Specifies the DEDB area name with up to eight alphanumeric characters starting with an alphabetic character. This name may be used as the label on the DBDGEN AREA statement. If a key file is specified for each area, this label must be used as the ddname of the corresponding key file.

This name must not be used as the label of the ANALYZE control statement or other AREADEF control statements.

This name must be a unique name in the ECSA of the user's system.

See "Preparation for using SR under IMS" on page 71 for the details of how this name is used by a DEDB SR generated with SRTYPE=MULTI specified.

UOW **(DEDB)**

This is a required parameter. The format is:

UOW=(n1,o1)

n1 Specifies the total number of CIs within a unit-of-work (UOW). The value must be a number between 2 and 32767.

o1 Specifies the number of CIs in the overflow area of a UOW.

The following condition must be satisfied:

$1 \leq o1 < n1$

Both n1 and o1 must be specified. This parameter has the same meaning as the UOW parameter on the DBDGEN control statement.

SIZE **(DEDB)**

Specifies the CI size (in bytes) of the DEDB to be created. This is a required parameter. The maximum value is 28672. The valid range conforms to IMS.

An efficient DEDB can be created by selecting the smallest CI size above the average record length.

Example:

1. Average record length is 200 bytes → SIZE=512
2. Average record length is 1000 bytes → SIZE=1024

ROOT **(DEDB)**

This is a required parameter. The format is:

```
ROOT=(n2,o2)
```

n2 Specifies the total number of UOWs per DEDB area. The value must be a number between 2 and 32767.

o2 Specifies the number of UOWs in the overflow area of a DEDB area.

The following condition must be satisfied:

$1 \le o2 < n2$

For DEDB, the total number of RAPs per area is calculated from UOW and ROOT parameters as follows:

$\text{RAP total} = (n1 - o1) \times (n2 - o2)$

The following condition must be satisfied if the maximum number of synonyms is specified with the OPTMZ parameter of the ANALYZE or AREADEF control statement:

$$(\text{RAP total} \times \text{Maximum number of synonyms}) \ge \text{Effective number of database keys}$$

This parameter is equivalent to the ROOT parameter of the DBDGEN control statement.

Note: The maximum number of RAP total that can be specified is 99999999.

TSIZE **(DEDB)**

Specifies the size of the equation table in bytes. This is a required parameter.

The number of equations should be in the range of 1 to 32767. The size of the equation table can be calculated as follows:

$$\text{TSIZE} = (\text{Number of equations}) \times (\text{Compressed key length} + 9)$$

The TSIZE value should be in the following range:

$$(\text{Compressed key length}) + 9 \le \text{TSIZE} \le (\text{Compressed key length} + 9) \times 32767$$

See Chapter 6, "Reference: Required amount of resources," on page 107 to determine the TSIZE value.

KEYRNG or ARETRY **(DEDB)**

Specifies the range of keys contained in an entry.

Either a KEYRNG parameter or an ARETRY parameter must be specified in an AREADEF control statement. KEYRNG and ARETRY are mutually exclusive.

Note: Consider using the KEYRNG parameter for new SRG JCLs. The ARETRY parameter is supported only for the compatibility with the existing SRG JCLs and with randomizers that were generated by using those JCLs. If you run a job that contains an ARETRY parameter, you might receive a FABO0212W message.

Explanations for the KEYRNG parameter in these topics also apply to the ARETRY parameter unless otherwise stated explicitly.

The syntax is:

KEYRNG=(*a*,*b*),(*a*',*b*').....

(*a*,*b*) *a* is the minimum value and *b* is the maximum value of the key range. The values need not be a value within the input key file.

A maximum of 3000 combinations of (*a*,*b*) can be specified. In addition, multiple (*a*,*b*) pairs can be used together with the later described AREXT parameter to specify one key range.

See also DELCHAR, CVCHAR, and AREXT parameter descriptions.

Example:

When the key is all numeric and the range of keys is 001000 to 001999 and 003000 to 003999, then:

- KEYRNG=(001000,001999),(003000,003999)

or

- KEYRNG=(001,001),(000,999),(003,003),(000,999) AREXT=2

a and *b* must be specified as a character or a hexadecimal number, depending on the attribute of the subkey field.

For the attributes N, I, A, C, Q, Y, or Z, the key range values should be specified in characters. One byte is occupied for each key byte.

For the attributes X, P, S, U, or O, the key range values should be specified in hexadecimal numbers. Two bytes are occupied for each key byte.

Note: The following symbols and a blank cannot be used for the key range values specified in characters:

& ' ( ) ,

Example:

If the key value specified on the ANALYZE control statement is

(1,2,C),(3,4,S),(5,6,N),(7,8,U,F0F1)

specify:

KEYRNG=(AA000C22F0F0,ZZ999C88F1F1)

AVRECL **(DEDB)**

Specifies the average database record length. This parameter must be specified if the DBLEN parameter of the ANALYZE statement is omitted. The maximum value is 65535. If both AVRECL and DBLEN parameters are specified, the AVRECL parameter is ignored.

Increase the size of this parameter to intentionally leave free space within CI.

Note: The prefix of each segment must be included in the database record length.

DELCHAR **(DEDB)**

Specifies the character to act as a delimiter between key range values for the KEYRNG parameter. This parameter is used to simplify KEYRNG parameter specification.

Example:

```
130 12345        KEYRNG=(130-12345,...)
 |      |        DELCHAR=-
Branch  Account
number  number
```

The specified character, which is the minus sign (-) in this example, is ignored as a key value within the KEYRNG parameter. Only one character can be specified.

CVCHAR **(DEDB)**

When the ANALYZE control statement specifies a key attribute that includes a blank, the character that is specified with the CVCHAR parameter is used to designate the blanks in the key range specification of the KEYRNG parameter.

Example:

□□12345 → KEYRNG=(@@12345,...),
CVCHAR=@

Only one character can be specified, and the specified character must not be equal to the character that is specified as DELCHAR.

AREXT **(DEDB)**

If the number of characters that are used to specify the minimum and maximum key range values for the KEYRNG parameter exceeds 127 bytes or if the specification on the KEYRNG parameter is extremely long, multiple (*a*,*b*) pairs can be used to specify one key range. The AREXT parameter specifies the number of these (*a*,*b*) pairs.

Example:

KEYRNG=(1234,1299),(56,99),(4567,4599),(89,99)
AREXT=2

is equivalent to:

KEYRNG=(123456,129999),(456789,459999)

AREXT parameter must be a number in the range of 1 - 4.

Default: 1

OPTMZ **(DEDB)**

Same as the OPTMZ parameter on the ANALYZE control statement. If the

OPTMZ parameter is specified on both the ANALYZE and AREADEF control statements, the one specified on the AREADEF control statement takes effect.

TLIMIT **(DEDB)**
Same as the TLIMIT parameter on the ANALYZE control statement.

TNUMLIM **(DEDB)**
Same as the TNUMLIM parameter on the ANALYZE control statement.

OPTMTIME **(DEDB)**
Same as the OPTMTIME parameter on the ANALYZE control statement.

LINKSR control statement

The LINKSR control statement is required only for generating DEDB SR. It specifies the names of DEDB areas to be combined.

The syntax for the LINKSR control statement is as follows:

```
LINKSR   AREAID=
        [,LINKID=]
        [,SRCCHK=]
```

AREAID **(DEDB)**

All DEDB areas must be specified with this parameter and the order of specification must be the same as the order of DBDGEN AREA statement. A maximum of 240 area names can be specified.

Example:

AREAID=AREA1,AREA2,AREA3

Note: SRG can generate a randomizer for DEDB that has up to 240 areas. DEDB that has more than 240 areas is not supported.

LINKID **(DEDB)**

Selects the areas to be assembled and link-edited.

ALL All the DEDB areas specified in the AREAID parameter are assembled and link-edited in order to generate a new SR load module.

(area1, area2, ...)

Specifies the DEDB areas to be assembled and link-edited. The areas must be predefined in the AREAID parameter. Only the selected areas are assembled and link-edited, and the current SR load module will be modified. Therefore, an SR load module must exist in the SR load module library when this option is specified.

Area names must be separated by each other with a comma and enclosed in parentheses, if multiple areas are specified.

sr_name

Specifies the name of the DEDB SR to be generated. Only the source file for the SRR part is reassembled and re-link-edited to modify the SRR part of the current SR load module. Any area equation table for the SR is neither re-assembled nor re-link-edited. When this option is specified, the current SR load module must exist in the SR load module library. If the SR load module type is MULTI, all modules for the SR must exist in the load module library. Furthermore, all source files for the SR must exist in the SR source library.

Default: ALL

SRCCHK **(DEDB)**

Specifies the level of the SR source file checking that must be done during the LINKSR process.

0 SRG does not check the SLOPL values and XPRI statements in the SRR and AREA SRT source files.

This is the default.

1 SRG checks, at LINKSR process, the consistency of the following:
- SLOPL values and the XPRI statements
- SLOPL value in the SRR source file and SLOPL values in all AREA SRT source files for the SR
- XPRI statements in the SRR source file and those in all AREA SRT source files for the SR.

The SLOPL values in the SRR are updated with the largest of all of the SRT SLOPL values.

If this option is specified, all AREA SRT source files must be in the SR source file library (ddname: SRRFILE). If your SR does not have many AREAs, you should specify SRCCHK=1 for the integrity check.

This is the recommended value.

2 This option is same as the option 1 except that the XPRI statements are rebuilt from the SRT source files if a missing XPRI statement is found in the SRR source file, or if the statement format is incorrect.

This option should be specified only when you find an error by specifying SRCCHK=1.

Default: SRCCHK=0

Notes:
1. SRCCHK=0 is the default, for compatibility with prior products.
2. SRCCHK=1 and SRCCHK=2 cannot be specified if you are migrating your old SR to the new format by specifying SRCMIG=YES parameter on the ANALYZE control statement.
3. Do not specify SRCCHK=2 unless you are getting an error with SRCCHK=1.

Database key file

A database key file is a data set that contains the key field of the root segment and the database record length of each database record in a database. It must be created before SRG execution.

The database key file can be created from an existing database or by extracting the keys from the root segment of a database to be generated.

The database key file has the following characteristics:

- It is a fixed-length blocked-record (FB) data set that contains the key fields of the root segment of the database to be created and optionally the database record length field.

  The database record length must include the length of the segment prefix because it is used for CI capacity ratio calculation. See *IMS Database Administration* for details concerning the segment prefix.
- The allowable maximum number of nonduplicate input keys in a key file for an HDAM database, a PHDAM partition, or a DEDB area is 16,777,215.
- The key field and the database record length field can be anywhere within a logical record. The key field must be a continuous area with a maximum length of 255 bytes, and the database record length field must be a 2-byte area.

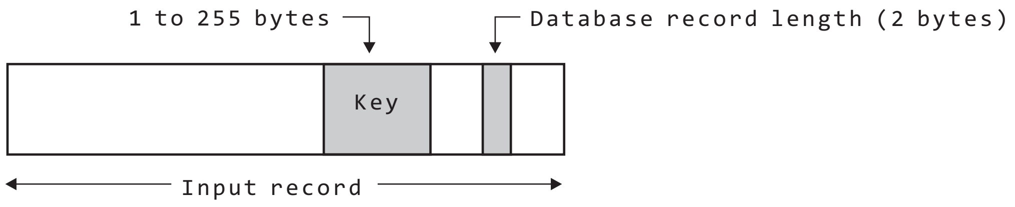


Figure 4. Logical Record of Database Key File

- In HDAM, the keys in the database key file are for the entire database.
- In PHDAM, the keys in the database key file are for the PHDAM partition for which the SR is to be used.
- In DEDB, the keys are either for the entire database (multiple areas), or for each DEDB area in the database.
- The keys must be sorted in ascending order.
- If the keys for multiple DEDB areas are included in one key file, the keys of an area might exist in separate parts in the key file (see Figure 6 on page 56).

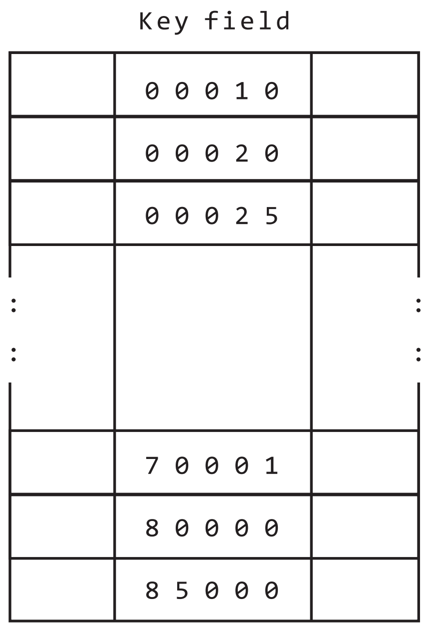


Figure 5. HDAM database key file

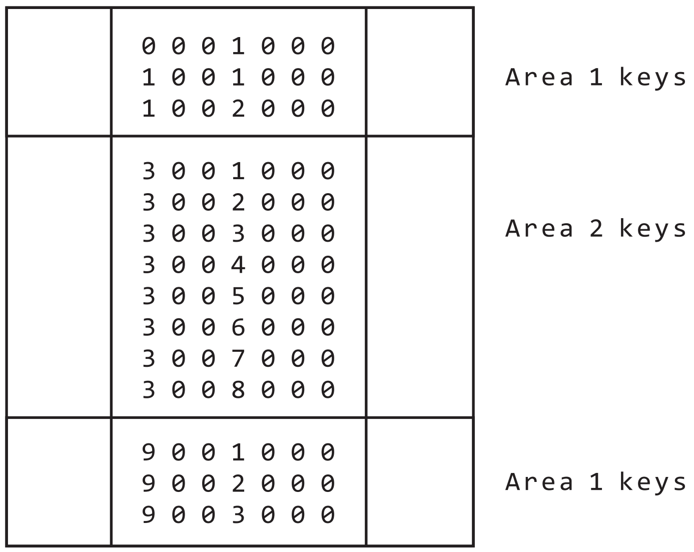


Figure 6. DEDB database key file (Key file for multiple areas)

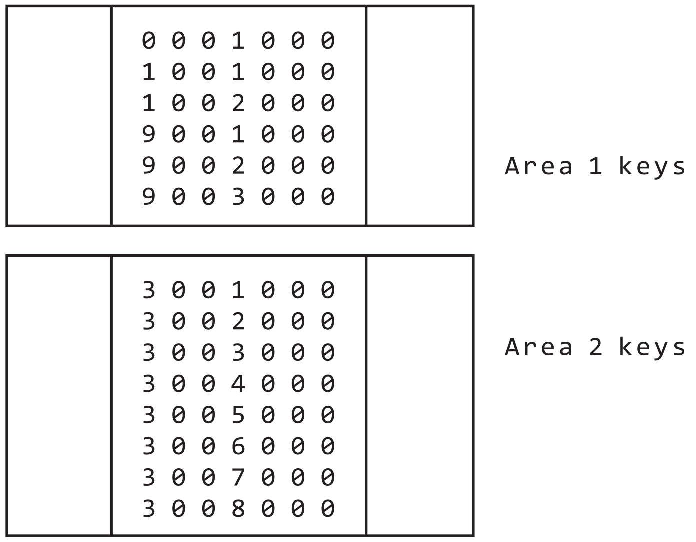


Figure 7. DEDB database key file (Key file for each area)

- A database key file can be created on DASD or magnetic tape. If on magnetic tape, it can have either a standard label (SL) or a nonstandard label (NL).
- The attributes of the key file must be specified in the control statements and in the parameters of the cataloged procedure (FABOSRG).
  – Attributes to be specified in the ANALYZE control statement:

Logical record length	INLRECL
Block length	INBLKSZ
Key field position	INOFFST
Key field length	INKEYL
Key field attribute	(a,b,c[,d])
Database record length position	DBLOFFST (for HDAM and PHDAM)DBLEN (for DEDB)

  – Attributes to be specified in the FABOSRG cataloged procedure parameters:

Data set name	INPTDSN
Volume serial number	SERIN
Unit	INUNIT
Label type and sequence number	NUM

  – Others

    The ddname of the key file is IN in FABOSRG.

    If there is only one key file, the above specifications are used. However, in the case of DEDB, if a key file is created for each area, a DD statement must be provided for each key file with the ddname matching the label on the corresponding AREADEF control statement. In addition, ddname IN must be specified as DD DUMMY, or INPTDSN='NULLFILE' must be specified in FABOSRG.

Considerations before running SRG

This topic describes some important issues that need careful consideration. Read the following descriptions carefully before running SRG.

See also Chapter 5, “Reference: How the Sequential Randomizer is generated,” on page 95 to understand how the SR is generated by SRG.

Database key file

SRG analyzes keys using only the database keys that are provided by the database key file. During IMS execution, if an attempt is made to insert a key that is not specified in the database key file, a synonym may occur. However, if the new key is known in advance, synonyms can be prevented by putting it in the database key file.

Although the database record length is optional, the sum of the database record length and segment prefix length should be placed in the database record length field because it is used to calculate the CI capacity ratio.

SRG examines the database key file in order to test the following:

- The database key has the same attributes as those specified in the ANALYZE control statement.
- There is no duplication or ascending sequence error of the database keys.
- The database record length matches the value that is specified in the database record length field. The record length field is checked only when DBLEN or DBLOFFST is specified.

If an error is found during the above check, an error message is issued. If SRG ends with RC=8 due to these errors, you must check the specifications on the ANALYZE control statement for the database key file or the key attributes, and rerun SRG with correct parameter values.

Key attributes

SRG compresses database keys based on the key attributes to improve key analysis efficiency. (See “Compressing database keys” on page 96.) In other words, unnecessary bits are dropped during grouping of the keys. Therefore, specify the correct key attribute in order to create an SR with fewer synonyms. Key attributes are specified with subkey field attribute parameters ((a,b,c) or (a,b,c,d)).

Key analysis method (Option 0, 1, 3)

When there are restrictions to the TSIZE or CORE specification, synonym occurrence depends on the key analysis method (option 0, 1, or 3). See Chapter 7, “Reference: Database key analysis methods,” on page 111 to determine the best option.

Equation table

In the database key analysis, the number of database key groups (the number of equations) is determined from the size of the equation table, which is specified by the TSIZE parameter of the AREADEF control statement (DEDB), or by the CORE parameter of the ANALYZE control statement (HDAM or PHDAM).

If the specified TSIZE (or CORE) is too small, a large number of synonyms may be generated if the key difference within a group is scattered.

If the maximum number of synonyms is specified for equation table optimization, equations that do not satisfy the condition are divided. Because the equation division is based solely on the number of synonyms, an equation with many synonyms is divided into many equations, and consequently, the size of the equation table becomes large regardless of the TSIZE (or CORE) value.

Therefore, the appropriate TSIZE (or CORE) value should be specified in order to prevent the equation table becoming too large at optimization.

The following steps describe how to fine-tune the TSIZE (or CORE) value:

1. Calculate the TSIZE (or CORE) value by referring to the formula in "How to estimate the size of resources" on page 110, and run SRG *without* specifying optimization. (If DEDB, do not specify LINKSR, and perform key analysis only. If HDAM or PHDAM, specify a temporary data set for IMSLIB in FABOSRG.)
2. Analyze the synonym occurrence by referring to the CI Distribution List report and Randomizer Statistics report, which are created by SRG after the key analysis. If many synonyms are clustered around a few RAPs or CIs (blocks), increase the TSIZE (or CORE) value and rerun SRG.

   Repeat this step until synonyms are evenly distributed. You may need to reconsider the subkey field attribute specifications or the optimization method (Option 0, 1, or 3).
3. Add the optimization specification and run SRG.

When the equation table is optimized, its size becomes large because equations are divided. If there are many synonyms before optimization, the equation table may become extremely large. Therefore, it is strongly recommended that you run SRG without optimization specification at first, and analyze the synonym occurrence and fine-tune the TSIZE (or CORE) size as described above.

If you want to avoid the great increase of the equation table during the optimization process, specify the TLIMIT option to limit the size of the equation table. Although the randomizing module generated this way may not be optimized to the point where it satisfies the synonym condition specified in the OPTMZ option, an equation table that is more optimized, within the TLIMIT, than the one created without OPTMZ option is created.

Work data set size

SRG uses work data sets during key analysis, equation table generation, and optimization. The sizes of the work data sets depend on the number and distribution of the keys. In the DEDB case, the sizes also depend on the load module type of the SR to be generated, and on the TYPECHG parameter. Determine the sizes by referring to the reference table in Chapter 6, "Reference: Required amount of resources," on page 107.

Sequential processing in multiple areas

In DEDB, if key ranges are not in ascending sequence among areas, the root segments may not be read in the ascending key sequence across areas when GN (get-next) calls are issued against the entire database. For example, if the key range of area 1 is 000 through 111 and 555 though 999, and the key range of area 2 is 222 through 444, the keys are read in the sequence of 000 through 111, 555 through 999, and 222 through 444 when GN calls are issued.

Compressed key file

You can create a compressed key file on storage by specifying the KEYNBR parameter of the ANALYZE control statement. The required storage size can be calculated as follows:

```
(INKEYL + CKEYL + 2) × KEYNBR
```

where INKEYL is the length of the input key that is specified in the INKEYL parameter of the ANALYZE control statement, and the CKEYL is the compressed-key length. Whether the compressed key file should be created on storage or on DASD depends on the environment in which SRG runs.

Size of the SR load module

The size of an SR load module is indicated by the message FABO0108I. If a DEDB SR is generated with SRTYPE=MULTI specified, the size indicated by the message is the total size of the modules composing the SR.

The optimization of the equation table may increase its size because the equations in the table are divided to satisfy the optimization conditions.

If the equation table is not optimized, SRG analyzes the keys based on TSIZE or CORE parameter value. However, if option 3 is specified, the size of the equation table can be increased up to twice the size specified with TSIZE or CORE parameter.

SR source library control

SRG stores SR sources into the SR source library. The SRR and SRT sources are used whenever the SR load module is regenerated (by LINKSR or SRGEN=YES). The SRR source is always updated whenever the SR load module is regenerated.

When you need to create an entirely new SR source from scratch (during testing, for example), first erase all the SRR and SRT sources, and then run SRG.

Note: This data set is important for SRG. It is recommended that you create backup copies regularly.

Module type of DEDB SR and type change function

A DEDB SR is to be generated usually as a single module. SRTYPE=MULTI should be specified only when the SR module size might exceed 16 MB.

It is recommended that the SR type not be frequently changed.

When it is necessary to change the module type of a DEDB SR, the user should make backup copies of the SR sources in SRRFILE (the source library of SR), and the load modules relating to the SR in IMSLIB (the load module library for SRs) before performing the type change by specifying TYPECHG=YES.

Extended call interface and DEDB Online Change

A DEDB SR is usually generated as a single module (SRTYPE=SINGLE) with the standard randomizer call interface (CALLTYPE=STD). You can change this type of DEDB SR online by using the DEDB Online Change function. See *IMS Database Administration* for information about the DEDB Online Change function.

If you want to generate a DEDB SR in the form of multiple modules (SRTYPE=MULTI) and change it online by use of the DEDB Online Change function, you *must* generate a randomizer that uses the Extended Call Interface (XCI). To do this, specify the parameter CALLTYPE=XCI in ANALYZE control statement. For more information on XCI SRs, see “Preparation for using SR under IMS” on page 71 and “How the

Sequential Randomizer (SR) works" on page 73. Only the SR with CALLTYPE=XCI can be defined as an XCI randomizer in DBDGEN.

Consider using the extended call interface when you generate a DEDB SR of SRTYPE=MULTI.

Using SR under DEDB Online Change environment

You should consider the following issues when using an SR under the DEDB Online Change environment.

Changing the SR online

If any change is made to the number of RAPs within each DEDB area or to the length of the database key, you must change the specifications of SRG control statements or the database key file (or both) accordingly and rerun SRG to generate an SR for the new database organization.

The DEDB SR determines the RAP number in two stages: firstly, it selects an area based on the root sequence key value and the key range definition that is specified by the KEYRNG parameters at the time of SR generation, then it calculates the RAP number within the selected area.

RAP number calculation for an area is based on the static equation table (Sequential Randomizer Table). Even when the number of RAPs is changed by a new ACB and when an area-level online change was performed to dynamically apply the new ACB, RAP number calculation by the SRG DEDB SR for that area is not affected by the new RAP definition. Thus, when the number of RAPs has been changed for an area by a new ACB, you must unload the old SR and load the new SR that has a new equation table for the changed area through a database-level online change. See "Restrictions" on page 11 for considerations when you define an SRG DEDB SR as a two-stage randomizer. To perform a database-level online change, follow the steps described in *IMS Database Administration*.

Consideration for the SR of SRTYPE=MULTI

If you are generating a DEDB SR of SRTYPE=MULTI and you want to change it online, you must generate it as a randomizer that uses the Extended Call Interface (XCI). To generate a DEDB SR of SRTYPE=MULTI as an XCI randomizer, you should specify CALLTYPE=XCI on the ANALYZE control statement. To define the randomizer as an XCI randomizer, you must code the RMNAME parameter of the DBD statement as follows in DBDGEN:

```
RMNAME=(xxxxxxxx,,,,XCI)
```

where *xxxxxxxx* is the name of the SR generated by SRG. How the XCI SR works is described in "How the Sequential Randomizer (SR) works" on page 73.

Output

SRG produces various types of reports, SRG control statement assembly listing, SR assembly listing, and SR link-edit listing. The user can specify the output reports and listings to be produced in the PRINTNG parameter of ANALYZE control statement.

The following table summarizes the reports and listings, and the condition of report/listing creation:

Table 5. Output reports/Listings

Output report/listing	Out to (DDNAME)	PRINTNG=			Condition	
		YES	NO	SHORT	HDAM	DEDB
CI Distribution List report	PRINT1	O	X	X	ANALYZE (SRGEN=NO)	AREADEF
Randomizer Statistics report	PRINT1	O	O	O	ANALYZE (SRGEN=NO)	AREADEF
Synonym Distribution Status report	PRINT1	O	O	O	ANALYZE (SRGEN=NO)	AREADEF
Recommended DBD Control Statement report	PRINT1	O	O	O	Always	LINKSR
DEDB Master Table report	PRINT1	O	O	O	N/A	LINKSR
SRG Control Statement Assembly Listing	SYSPRINT	O	O	O	Always	Always
SR Assembly Listing • SRR part • SRT part • DBDGEN control statements	SYSPRINT	O	O (*)	X	Always	LINKSR
SR Link-edit Listing	SYSPRINT	O	O	O	Always	LINKSR

Note: O indicates that the report or listing is to be produced. X indicates that the report or listing is not to be produced. For (*), see the PRINTNG parameter description in PRINTNG.

The following five types of reports are described in detail in the following topics:
- "CI Distribution List report" on page 63
- "Randomizer Statistics report" on page 65
- "Synonym Distribution Status report" on page 66
- "Recommended DBD Control Statement report" on page 68
- "DEDB Master Table report" on page 70

CI Distribution List report

This report shows the minimum and maximum keys for each CI (block), number of synonyms, CI capacity ratio, and maximum number of synonyms per RAP. It also shows the number of keys and synonyms for each UOW (cylinder).

Use this report to determine the database key and synonym distribution. If synonyms are clustered or if the CI capacity ratio is high, rerun SRG specifying the OPTMZ parameter to improve the performance and efficiency of SR.

This report is produced on the following conditions:

For HDAM and PHDAM
When PRINTNG=YES and SRGEN=NO is specified for the ANALYZE control statement

For DEDB
When PRINTNG=YES is specified for the ANALYZE control statement, and AREADEF control statement is specified.

Figure 8 on page 64 shows the sample output of CI Distribution List report. This report contains the following information:

NAME
In HDAM and PHDAM, this field shows the name of the SR. In DEDB, this field shows the name of the area.

INPUT KEY LENGTH
This field shows the length of the input key.

COMPRESSED KEY LENGTH
This field shows the length of the compressed key.

BLOCK NO
This field shows the CI (block) number calculated by SR.

LOW KEY
This field shows the smallest key assigned to this CI (block).

HIGH KEY
This field shows the largest key assigned to this CI (block).

NBR OF KEYS
This field shows the number of keys assigned to this CI (block).

SYNONYM
This field shows the total number of synonyms in this CI (block).

CAPACITY(%)
This field shows the CI (block) utilization ratio. It is calculated as follows:

(Sum of database record length / CI (block) size) x 100

MAX SYNONYM
This field shows the maximum number of synonyms per RAP in this CI (block).

<xxxxxxxx>
This field shows the relative UOW (cylinder) number.

TOTAL KEY
This field shows the total number of keys per UOW (cylinder).

TOTAL SYNONYM
This field shows the total number of synonyms per UOW (cylinder).

```
IMS SEQUENTIAL RANDOMIZER GENERATOR                   "CI DISTRIBUTION LIST REPORT"                        PAGE:     1
5655-E11                                             DATE: 06/01/2014  TIME: 21.41.47                     FABOMAIN - V1.R1

         NAME: SRGSR1    INPUT KEY LENGTH: 00200  COMPRESSED KEY LENGTH: 00146

         BLOCK NO        LOW KEY             HIGH KEY        NBR OF KEYS          SYNONYM       CAPACITY(%)           MAX SYNONYM
         --------        --------            --------        -----------          -------       -----------           -----------
         00000001        * THIS BLOCK IS NOT RANDOMIZED
         00000002        040480C0E0...       040480C0E0...      00013              00000           00095                 00000
         00000003        040480C0E0...       040480C0E0...      00013              00000           00095                 00000
         00000004        040480C0E0...       040480C0E0...      00013              00000           00095                 00000
         00000005        040480C0E0...       040480C0E0...      00013              00000           00095                 00000
         00000006        040480C0E0...       040480C0E0...      00013              00000           00095                 00000
         00000007        040480C0E0...       040480C0E0...      00013              00000           00095                 00000
         00000008        040480C0E0...       040480C0E0...      00013              00000           00095                 00000
         00000009        040480C0E0...       040480C0E0...      00009              00000           00065                 00000
         00000010        09090E0703...       09090E0703...      00017              00010           00124                 00010
         00000011        09090E0703...       09090E0703...      00013              00000           00095                 00000
         00000012        09090E0703...       09090E0703...      00013              00000           00095                 00000
         00000013        09090E0703...       09090E0703...      00013              00000           00095                 00000
         00000014        09090E0703...       09090E0703...      00013              00000           00095                 00000
         00000015        09090E0703...       09090E0703...      00013              00000           00095                 00000
         00000016        09090E0703...       09090E0703...      00013              00000           00095                 00000
         00000017        09090E0703...       09090E0703...      00005              00002           00036                 00002
         00000018        0D0D1E0F87...       0D0D1E0F87...      00021              00010           00153                 00010
         00000019        0D0D1E0F87...       0D0D1E0F87...      00013              00000           00095                 00000
         00000020        0D0D1E0F87...       0D0D1E0F87...      00013              00000           00095                 00000
         00000021        0D0D1E0F87...       0D0D1E0F87...      00013              00000           00095                 00000
         00000022        0D0D1E0F87...       0D0D1E0F87...      00013              00000           00095                 00000
         00000023        0D0D1E0F87...       0D0D1E0F87...      00013              00000           00095                 00000
         00000024        0D0D1E0F87...       0D0D1E0F87...      00013              00000           00095                 00000
         00000025        0D0D1E0F87...       12120080C0...      00013              00010           00095                 00010
         00000026        12120080C0...       12120080C0...      00013              00000           00095                 00000
         00000027        12120080C0...       12120080C0...      00013              00000           00095                 00000
         00000028        12120080C0...       12120080C0...      00013              00000           00095                 00000
         00000029        12120080C0...       12120080C0...      00013              00000           00095                 00000
         00000030        12120080C0...       12120080C0...      00013              00000           00095                 00000
              .
              .
              .
         00000044        1B1B3C1E0F...       1B1B3C1E0F...      00013              00000           00095                 00000
         00000045        1B1B3C1E0F...       1B1B3C1E0F...      00013              00000           00095                 00000
         00000046        1B1B3C1E0F...       1B1B3C1E0F...      00013              00000           00095                 00000
         00000047        1B1B3C1E0F...       1B1B3C1E0F...      00015              00008           00109                 00008
         00000048        2020010080...       2020010080...      00011              00010           00080                 00010
         00000049        2020010080...       2020010080...      00013              00000           00095                 00000
         00000050        2020010080...       2020010080...      00013              00000           00095                 00000
         00000051        2020010080...       2020010080...      00013              00000           00095                 00000
         00000052        2020010080...       2020010080...      00013              00000           00095                 00000
         00000053        2020010080...       2020010080...      00013              00000           00095                 00000
         00000054        2020010080...       2020010080...      00013              00000           00095                 00000
        <00000000>       * UOW OR CYLINDER HAS CHANGED *    TOTAL KEY:00000700  TOTAL SYNONYM:00000075
         00000055        2020010080...       2020010080...      00011              00000           00080                 00000
```

Figure 8. CI Distribution List report (HDAM and PHDAM)

Note: In HDAM and PHDAM, the CI (block) number that is specified by SR starts from 1, but the CI (block) number 1 which contains a bit map is *not* randomized. Also, the RAPs of the second and subsequent bit map CIs or blocks are always not randomized because they are not used.

In DEDB, the CI (block) number starts from 0 because the SR calculates the relative RAP number (relative CI number) within the area.

Randomizer Statistics report

This report shows the SR execution time, CI utilization, number of synonyms, and synonym distribution.

This report is produced on the following conditions:

For HDAM and PHDAM
When SRGEN=NO is specified for the ANALYZE control statement (regardless of the PRINTNG option)

For DEDB
When the AREADEF control statement is specified (regardless of the PRINTNG option).

See the following figure for the sample output of this report for an HDAM and PHDAM database.

```
IMS SEQUENTIAL RANDOMIZER GENERATOR                       "RANDOMIZER STATISTICS REPORT"                          PAGE:     1
5655-E11                                                 DATE: 06/01/2014  TIME: 21.41.47                       FABOMAIN - V1.R1


             ***   RANDOMIZER STATISTICS INFORMATION FOR  SRGSR1    ***


          MAXIMUM SYNONYMS IN UOW(CYLINDER) ARE                        00000075
          TOTAL SYNONYMS IN FILE ARE                                   00000075
          TOTAL KEYS     IN FILE ARE                                   00000700
          THE AVERAGE  EXECUTION TIME IN THE SR IS                    0.0004626 SEC
          THE SHORTEST EXECUTION TIME IN THE SR IS                    0.0002650 SEC
          THE LONGEST  EXECUTION TIME IN THE SR IS                    0.0155218 SEC
          TOTAL NO. OF CIS(BLOCKS)     IN THE DATA BASE IS             00000055
          TOTAL NO. OF FREE CI(BLOCKS) IN THE DATA BASE IS             00000001
          THE LAST USED CI (BLOCK) NO. IN THE DATA BASE IS             00000055
          NO. OF RAPS / CI (BLOCK) IS                                  00000013
```

Figure 9. Randomizer Statistics report (HDAM and PHDAM)

The report contains the following statistical information:

- SR name (HDAM and PHDAM) or area name (DEDB)
- Maximum number of synonyms in UOW (cylinder)
- Total number of synonyms in the database
- Total number of input keys
- Maximum, minimum, and average time of SR execution

  (These are the data produced by the SR simulation; the actual data may vary depending on the user's IMS environment in which the SR is executed.)
- Total number of CIs in the database and the number of unused CIs
- The last CI number used
- Number of RAPs per CI. In DEDB, this value is always 1.

Synonym Distribution Status report

This report shows the database key distribution based on the number of synonyms in a RAP or CI (block).

This report is produced on the following conditions:

For HDAM and PHDAM
: When SRGEN=NO is specified for the ANALYZE control statement (regardless of the PRINTNG option)

For DEDB
: When the AREADEF control statement is specified (regardless of the PRINTNG option)

```
IMS SEQUENTIAL RANDOMIZER GENERATOR                      "SYNONYM DISTRIBUTION STATUS REPORT"                            PAGE:     1
5655-E11                                                   DATE: 06/01/2014  TIME: 21.41.47                         FABOMAIN - V1.R1


 DISTRIBUTION PER ANCHOR POINT   ******  DISTRIBUTIONS PER BLOCK           **
 #CHAINS   ROOTS CUMROOT PERCTNG *DIST*  #BLOCKS * ROOTS  CUMROOT  PERCTNG  **
                                 ******
     616     616     616   88.0% *   1*                                .0%  **
                     616   88.0% *   2*                                .0%  **
       1       3     619   88.4% *   3*                                .0%  **
                     619   88.4% *   4*                                .0%  **
                     619   88.4% *   5*       1        5        5      .7%  **
       1       6     625   89.2% *   6*       1        6       11     1.5%  **
                     625   89.2% *   7*                         11     1.5%  **
                     625   89.2% *   8*                         11     1.5%  **
       1       9     634   90.5% *   9*       1        9       20     2.8%  **
                     634   90.5% *  10*       1       10       30     4.2%  **
       6      66     700  100.0% *  11*       2       22       52     7.4%  **
                     700  100.0% *  12*                         52     7.4%  **
                     700  100.0% *  13*      43      559      611    87.2%  **
                     700  100.0% *  14*                        611    87.2%  **
                     700  100.0% *  15*       1       15      626    89.4%  **
                     700  100.0% *  16*       1       16      642    91.7%  **
                     700  100.0% *  17*       1       17      659    94.1%  **
                     700  100.0% *  18*                        659    94.1%  **
                     700  100.0% *  19*                        659    94.1%  **
                     700  100.0% *  20*       1       20      679    97.0%  **
                     700  100.0% *  21*       1       21      700   100.0%  **

 TOTAL RAP USED      625  TOTAL BLOCKS USED           54
```

Figure 10. Synonym Distribution Status report (HDAM and PHDAM)

The report contains the following information on synonym distribution:

DISTRIBUTION PER ANCHOR POINT
: This field shows the key distribution based on the number of synonyms in a RAP.

  DIST This field shows the number of keys assigned to a RAP (same as the number of synonyms per RAP plus 1).

  #CHAINS
  : This field shows the number of used RAPs to which *nn* keys (*nn* = DIST value) were assigned.

  ROOTS
  : This field shows the total number of keys assigned to all the RAPs to which *nn* keys (*nn* = DIST value) were assigned.

    The value of this field is calculated as follows:

ROOTS = #CHAINS x DIST

CUMROOT
This field shows the cumulative number of ROOTS.

PERCTNG
This field shows the percentage of CUMROOT in the total number of keys. The value is calculated as follows:

PERCTNG = (CUMROOT / Total number of keys) x 100

DISTRIBUTIONS PER BLOCK
This field shows the key distribution based on the number of synonyms in a CI (block).

DIST This field shows the number of keys assigned to a CI (block). It is the same as the number of synonyms per CI (block) plus 1.

#BLOCKS
This field shows the number of used CIs (blocks) to which *nn* keys (*nn* = DIST value) were assigned.

ROOTS
This field shows the total number of keys assigned to all the CIs (blocks) to which *nn* keys (*nn* = DIST value) were assigned. The value is calculated as follows:

ROOTS = #BLOCKS x DIST

CUMROOT
This field shows the cumulative number of ROOTS.

PERCTNG
This field shows the percentage of CUMROOT in the total number of keys. The value is calculated as follows:

PERCTNG = (CUMROOT / Total number of keys) x 100

TOTAL RAP USED
This field shows the total number of RAPs used by the SR. The value is equivalent to the sum of #CHAINS.

TOTAL BLOCKS USED
This field shows the total number of blocks used by the SR. The value is equivalent to the sum of #BLOCKS.

Note: In case of DEDB, the RAP distribution and CI distribution are the same because there is only one RAP per CI.

Recommended DBD Control Statement report

This report contains the information on DBDGEN control statements that are required for running the generated SR. Use this information to specify DBDGEN control statements.

This output report is produced on the following conditions:

For HDAM and PHDAM
: This report is always produced, regardless of the PRINTNG option.

For DEDB
: This report is produced only when the LINKSR control statement is specified, regardless of the PRINTNG option. If the LINKSR control statement is not specified, the UOW and ROOT parameters are shown in the message FABO1301A only when the number of RAPs required by SRG has been changed.

When PRINTNG=YES or PRINTNG=NO, the DBDGEN control statements are also shown in the SR assembly listing (ddname: SYSPRINT). If there are multiple DEDB areas, DBDGEN control statements are produced for each area in the ddname SYSPRINT.

See Figure 11 and Figure 12 on page 69 for the sample output of this report:

HDAM and PHDAM

```
IMS SEQUENTIAL RANDOMIZER GENERATOR                "RECOMMENDED DBD CONTROL STATEMENT REPORT"                  PAGE:     1
5655-E11                                                DATE: 06/01/2014  TIME: 21.41.47                      FABOMAIN - V1.R1

                                    *----------------------------------------------------------*
                                    *                                                          *
                                    *    RECOMMENDED DBD STATEMENTS FOR SR : SRGSR1            *
                                    *                                                          *
                                    *----------------------------------------------------------*


                                          DBD     NAME=(---),ACCESS=(HDAM,OSAM),                           X
                                                  RMNAME=(SRGSR1,013,00000056,---)
                                          DATASET DD1=---,DEVICE=-----,SIZE=04096
                                          SEGM    NAME=---,PARENT=0,BYTES=---,PTR=TB
                                          FIELD   NAME=(---,SEQ,U),BYTES=200,START=---,TYPE=X
```

Figure 11. Recommended DBD Control Statement report (HDAM and PDAM)

This report contains the following information:

- Database type (VSAM or OSAM)
- Name of the SR generated by SRG
- Number of RAPs/CI (or block)
- Total number of CIs (or blocks)
- CI (Block) size
- Root segment key length

For PHDAM SR, this report should be read as follows:

- The first operand of the ACCESS parameter is shown as HDAM in the report. You must, however, specify PHDAM for the ACCESS parameter at DBDGEN.

  If the above example is for PHDAM, you must code as follows:

  ```
  ACCESS=(PHDAM,OSAM)
  ```

- The operands of the RMNAME parameter—the name of the randomizer, the number of RAPs per CI or block, and the total number of CIs or blocks in the root addressable area—are to be defined with the HALDB Partition Definition Utility.
- The DATASET statement is not specified in the DBDGEN but is specified by the DSGROUP parameter in the SEGM statement. The CI size or the block size for the data set must be defined with the HALDB Partition Definition Utility.

DEDB

```
IMS SEQUENTIAL RANDOMIZER GENERATOR                  "RECOMMENDED DBD CONTROL STATEMENT REPORT"                   PAGE:     1
5655-E11                                                   DATE: 06/01/2014  TIME: 21.22.49                      FABOMAIN - V1.R1

                             *-----------------------------------------------------------*
                             *                                                           *
                             *    RECOMMENDED DBD STATEMENTS FOR SR : SRGSR2             *
                             *                                                           *
                             *     DBD NAME=(---),ACCESS=DEDB,RMNAME=SRGSR2              *
                             *                                                           *
                             *-----------------------------------------------------------*

         ------------------------------------------------------------------------------------------------------------
             AREA NAME    SIZE PARAMETER    UOW PARAMETER        ROOT PARAMETER      TOTAL NUMBER OF RAPS
                                                                                     IN ROOT ADDRESSABLE AREA
         ------------------------------------------------------------------------------------------------------------


            SRGSR21      SIZE=08192        UOW=(00070,00020)    ROOT=(00100,00020)         00004000
            SRGSR22      SIZE=00512        UOW=(00070,00020)    ROOT=(00100,00020)         00004000
            SRGSR23      SIZE=00512        UOW=(00051,00001)    ROOT=(00061,00001)         00003000
            SRGSR24      SIZE=00512        UOW=(00051,00001)    ROOT=(00061,00001)         00003000
```

Figure 12. Recommended DBD Control Statement report (DEDB)

This report contains the following information:

RMNAME
This field shows the name of SR generated by SRG.

AREA NAME
This field shows the DEDB area name.

SIZE PARAMETER
This field shows the CI size.

UOW PARAMETER
This field shows the UOW parameter value specified in DBDGEN.

UOW = (Total number of UOWs per area, Number of overflow UOWs per area).

ROOT PARAMETER
This field shows the ROOT parameter value specified in DBDGEN.

ROOT = (Total number of CIs per UOW, Number of overflow CIs per UOW).

TOTAL NUMBER OF RAPS IN ROOT ADDRESSABLE AREA
This field shows the total number of RAPs in the root addressable area calculated from the UOW and ROOT values.

DEDB Master Table report

The following figure shows the DEDB Master Table report, which describes the key range of each area, specified by the KEYRNG or ARETRY parameter of the AREADEF control statement, as a table for each database.

This report is produced only for DEDB databases when the LINKSR control statement is specified (regardless of the PRINTNG option).

```
IMS SEQUENTIAL RANDOMIZER GENERATOR                       "DEDB MASTER TABLE REPORT"                          PAGE:     1
5655-E11                                              DATE: 06/01/2014  TIME: 21.22.49                       FABOMAIN - V1.R1

**RANDOMIZING MODULE=SRGSR2

        +-------------------------------------------------------------------------------------------------------------+
        |                  LOW KEY                         |                HIGH KEY                         | AREA NAME |
        +-------------------------------------------------------------------------------------------------------------+
        | 000000000000000C                                 | 0BE0FFFF9999999C                                |  SRGSR21  |
        +-------------------------------------------------------------------------------------------------------------+
        | 0BE100000000000C                                 | 1737FFFF9999999C                                |  SRGSR22  |
        +-------------------------------------------------------------------------------------------------------------+
        | 173800000000000C                                 | 1CEFFFFF9999999C                                |  SRGSR23  |
        +-------------------------------------------------------------------------------------------------------------+
        | 1CF000000000000C                                 | FFFFFFFF9999999C                                |  SRGSR24  |
        +-------------------------------------------------------------------------------------------------------------+
```

Figure 13. DEDB Master Table report

The report contains the following information:

RANDOMIZING MODULE
This field shows the name of the SR generated by SRG.

LOW KEY
This field shows the smallest key in the equation table for the DEDB area (shown in hexadecimal numbers or in characters depending on the key attribute).

HIGH KEY
This field shows the largest key in the equation table for the DEDB area (shown in hexadecimal numbers or in characters depending on the key attribute).

AREA NAME
This field shows the name of the DEDB area.

Preparation for using SR under IMS

Before you use an SR under IMS, you must consider certain factors.

DBDGEN and HALDB partition definition

From the result of the key analysis, SRG produces a report of the recommended DBDGEN control statement.

The DBDGEN control statements are listed either on the SR assembly listing (ddname: SYSPRINT) or on ddname PRINT1. See "Recommended DBD Control Statement report" on page 68. To perform DBDGEN correctly, these control statements should be referred to.

Change the specifications of the SRG control statements or the database key file according to whether the root addressable area or the length of the database key is changed. Then re-create the SR by running the SRG job again.

If the DBDGEN control statements produced by SRG differ from the existing DBDGEN, you must run DBDGEN for the database before running the SR generated by SRG. For an SR for a partition of a PHDAM database, you must use the HALDB Partition Definition Utility, in addition to the DBDGEN utility, to define or change the randomizer parameters or the CI (or block) size.

After DBDGEN (and HALDB Partition Definition for PHDAM), test the result and check that the DBD and the SR specifications match. If not, SR issues the message FABO0001A during the IMS execution.

Work area storage

A DEDB SR, or a reusable or reentrant SR for HDAM or PHDAM, allocates a 2 KB work area in ECSA for each dependent region in which the SR is used. This work area is reused when a DL/I call for a database that uses an SR of one of the above types is issued from the dependent region.

Control blocks for non-XCI DEDB SR of SRTYPE=MULTI

Non-XCI DEDB SRs of the multiple-module type use the following control blocks:

- 80 bytes in CSA for the BREESCD table (the extended ESCD for the SRG)
- 14,696 bytes in ECSA for the SDOTABLE (the load control table for the DEDB SRs of multiple module type)
- 2,408 bytes in ECSA for the SDOTABO1 (the overflow table of the SDOTABLE).

These blocks are allocated by a non-XCI DEDB SR of the multiple-module type at first call to the SR, if another non-XCI DEDB SR of the multiple-module type has not acquired them previously. These control blocks are commonly used for all non-XCI DEDB SRs of the multiple-module type.

Because only the SRR (Sequential Randomizing Routine) module is loaded by IMS, SRR loads the extended master table and all area equation tables for the SR into ECSA. (For the structure of a DEDB SR of the multiple-module type, see "Generating an SR load module" on page 105.) Thus, if a non-XCI DEDB SR of the multiple-module type is used, it is highly recommended that, after IMS starts, you run an application program that issues at least one GU DL/I call for the database for which the SR is used.

Tip: XCI DEDB SRs of the multiple-module type do not use these control blocks. If no non-XCI DEDB SR is used in your IMS system, these control blocks are not allocated. Consider using the XCI interface (specifying CALLTYPE=XCI on the ANALYZE control statement) when you generate a DEDB SR of the multiple-module type.

The size of an SR

The size of the SR is shown in the message FABO0108I when the SR is generated by SRG. For DEDB SR generated with SRTYPE=MULTI specified, this is the total size of all the modules for the SR—that is, the total size of the following:

- SRR module of the SR
- Extended master table for the SR
- Area equation table of all areas for the SR

Note: For a non-XCI SR, these areas are occupied and reused while IMS is up and running. They are released when IMS is shut down. For an XCI SR, these areas are allocated when a database for which the SR is used is started at the initialization of IMS or when the database is started by a /START DB command, and the areas are released when /DBR DB commands are issued for all databases for which the SR is used or IMS is shut down.

How the Sequential Randomizer (SR) works

The following topics describe how the Sequential Randomizer (SR) works under IMS, and the errors that might occur.

The SR is loaded by IMS, and is invoked each time an application program issues a DL/I call to the database or PHDAM partition for which the SR is used.

- “How the SR works for a DEDB”
- “How the SR works for an HDAM or PHDAM database” on page 76
- “Abend” on page 77
- “SR action for an unexpected key” on page 77

How the SR works for a DEDB

How the SR works for a DEDB database depends on whether the SR is generated as a single module (SRTYPE=SINGLE) or as multiple modules (SRTYPE=MULTI).

DEDB SR of SRTYPE=SINGLE

This is the default type of the DEDB SR.

At initialization, IMS loads the SR directly into the extended CSA (ECSA)—CSA above the 16 MB line. SR does the following:

1. Allocates a 2 KB work area for the IMS dependent region from which the call was issued, the area has not yet been allocated.

   Exception: When you use the SR with certain utilities of IMS Fast Path Solution Pack or with the IMS MSDB-to-DEDB Conversion utility (DBFUCDB0), the SR is loaded into the extended private area and the work area is allocated in the extended private area. The same exception applies when you call the SR from your own program that runs in the MVS batch environment.

   Requirement: When you call an SR from your own program that runs in the MVS batch environment, you must use the z/OS batch utility interface that is defined by IMS for DEDB randomizers. For more information about DEDB randomizers and how IMS communicates with such randomizers, see the topic "Sample data entry database randomizing routines (DBFHDC40 / DBFHDC44)" in *IMS Exit Routines*.

   For an SRG SR, in addition to this requirement, you must initialize the eight-word area with binary zeros before the first call is made to the SR. The address of this eight-word area is passed to the DEDB randomizer by using register 6 at each call. This eight-word area contains the pointer to the SR's work area, which is allocated at the first call. Also, you must ensure that the eight-word area is reused in all succeeding calls and that the content of the eight-word area is not modified by any program other than the SR.
2. Compresses the input database keys.
3. Uses the compressed keys to calculate the DMAC address and relative RAP number within the area.

DEDB SR of SRTYPE=MULTI

A DEDB SR of SRTYPE=MULTI is generated as multiple modules. This type of DEDB SR can have the standard call interface (CALLTYPE=STD) or the extended call interface (CALLTYPE=XCI).

Recommendation: Consider using the XCI interface (CALLTYPE=XCI) when you generate a DEDB SR of the multiple-module type.

CALLTYPE=STD

IMS loads the SR directly into ECSA at initialization; only the SRR module is loaded by IMS.

An SR of this type does the following:

1. Allocates a 2 KB work area for the IMS-dependent region from which the call was issued, if it has not yet been allocated.

   The SR also allocates the following areas at the time of the first DL/I call, if another SR has not acquired them previously:

   - BREESCD table
   - SDOTABLE
   - SDOTABO1, if needed

   For the size of these areas, see "Preparation for using SR under IMS" on page 71. After these areas are allocated, SR loads the extended Master table and all area equation tables for the SR either into ECSA.

   Exception: When you use the SR with certain utilities of IMS Fast Path Solution Pack or with the IMS MSDB-to-DEDB Conversion utility (DBFUCDB0), the SR, the extended master table, and all area equation tables are loaded into the extended private area and the work area is allocated in the extended private area. The same exception applies when you call the SR from your own program that runs in the MVS batch environment.

   Requirement: When you call an SR from your own program that runs in the MVS batch environment, you must use the z/OS batch utility interface that is defined by IMS for DEDB randomizers. For more information about DEDB randomizers and how IMS communicates with such randomizers, see the topic "Sample data entry database randomizing routines (DBFHDC40 / DBFHDC44)" in *IMS Exit Routines*.

   For an SRG SR, in addition to this requirement, you must initialize the eight-word area with binary zeros before the first call is made to the SR. The address of this eight-word area is passed to the DEDB randomizer by using register 6 at each call. This eight-word area contains the pointer to the SR's work area, which is allocated at the first call. Also, you must ensure that the eight-word area is reused in all succeeding calls and that the content of the eight-word area is not modified by any program other than the SR.
2. Compresses the input database keys.
3. Uses the compressed keys to calculate the DMAC address and relative RAP number within the area.

CALLTYPE=XCI

At the initialization of IMS, or when a /START DB command is processed, IMS first loads the SR into ECSA and then makes an initialization call to invoke the initialization routine of the SR. Within the SR initialization routine, the SR loads the extended master table and all area equation tables for the SR into ECSA. It also saves the address of the extended master table in the userdata field passed from IMS. At each regular randomizing call, the address of the extended master table for the SR is obtained from the userdata field.

If the SR is used by a batch caller, the batch caller must first load the SR into the extended private area and then make an initialization call to invoke the initialization routine of the SR. Within the SR initialization routine, the SR loads the extended master table and all area equation tables for the SR into the extended private area. The only difference in call parameters is the value that must be set in the caller environment field.

Note: See *IMS Exit Routines* for details of the call parameters, including the userdata, for XCI randomizers.

If an error occurs during the initialization call from an online IMS subsystem, the following IMS message is issued:

DFS2627I
: **ERROR IN INITIALIZING XCI RANDOMIZER, *rrrrrrrr*, FOR DEDB=*dddddddd*, RETURN CODE=*xxxx*, REASON CODE=*yyyyyyyy***

If an error occurs during the initialization call from a batch caller, the FABO0010A or FABO0011A message (SR message) is issued.

For a description of the SR return codes and reason codes, see "SR return codes and reason codes" on page 131.

Note: Since all the tables are loaded at the time of the initialization call, it is not necessary to run an application program, as is necessary for a DEDB SR of SRTYPE=MULTI and CALLTYPE=STD. (See "Preparation for using SR under IMS" on page 71.)

While a /DBRECOVERY DB command is being processed, IMS issues a termination call to the SR to invoke its termination routine, and then unloads the SR. In the SR's termination routine, the SR deletes all area equation tables and the extended master table for the SR. Then the userdata field that is passed from IMS and contains the address of the extended master table is cleared.

Important: If the SR is invoked from a batch caller, the batch utility must issue a termination call to the SR to invoke its termination routine, and then unload the SR. When the termination call is issued, the userdata field that is returned from the initialization call must be set for the userdata field of the parameter list. The termination call must be issued by the same task that issued the initialization call.

If an error occurs during the termination call from an online IMS subsystem, the following IMS message is issued:

DFS2628I
: **ERROR IN TERMINATING XCI RANDOMIZER, *rrrrrrrr*, FOR DEDB=*dddddddd*, RETURN CODE=*xxxx*, REASON CODE=*yyyyyyyy***

If an error occurs during the termination call from a batch caller, the FABO0010A or FABO0011A message (SR message) is issued.

For a description of the SR return codes and reason codes, see "SR return codes and reason codes" on page 131.

The code used in determining the DMAC address and the relative RAP number from the key is the same as that for CALLTYPE=STD. SR does the following:

1. Allocates a 2 KB work area for the IMS dependent region from which the call was issued, if it has not yet been acquired.
2. Compresses the input database keys.
3. Uses the compressed keys to calculate the DMAC address and relative RAP number within the area.

Exception: When the SR is used by certain utilities of IMS Fast Path Solution Pack or MVS batch programs, the SR, the extended master table, and all area equation tables are loaded into the extended private area and the work area is also allocated in the extended private area.

Requirement: When you call an SR from your own program that runs in the MVS batch environment, you must use the enhanced call parameter list for XCI randomizers. This enhanced call parameter list was introduced by APAR PK40256 for IMS Version 10, and is supported in later versions of IMS. The SR expects that your MVS batch program sets a four-byte character string 'OS ' in the caller environment field of the call parameter list that is used by IMS XCI initialization, randomizing, and termination calls. The SR uses this field to determine whether the caller runs under IMS online or MVS batch. For more information about DEDB XCI randomizers and how IMS communicates with such randomizers, see the topic "Extended call interface (XCI) option" in *IMS Exit Routines*.

For an SRG SR, in addition to this requirement, you must initialize the eight-word work area with binary zeros before the first randomizing call is made to the SR. The address of this eight-word work area is passed to the DEDB randomizer at every randomizing call by using the call parameter list. This eight-word work area contains the pointer to the SR's work area, which is allocated at the first randomizing call. Also, you must ensure that the eight-word work area is reused in all succeeding randomizing calls and that the content of the eight-word work area is not modified by any program other than the SR.

Note: The IMS MSDB-to-DEDB Conversion utility (DBFUCDB0) does not support this batch interface for XCI randomizers. Therefore, SR XCI randomizers cannot be used with the IMS MSDB-to-DEDB Conversion utility.

How the SR works for an HDAM or PHDAM database

An HDAM SR is loaded by IMS when the database is opened. How the SR works for an HDAM database depends on the attribute of the SR; an HDAM SR can have one of the following attributes:

Nonreusable

IMS serializes the database before calling the SR. If the SR is used for multiple databases, each database has its own copy of the SR.

The nonreusable attribute is the default attribute for the HDAM SR generated by SRG and is compatible with HDAM SRs created by DBT Version 2 SDO.

Reusable
: IMS serializes the database before calling the SR. If the SR is used for multiple databases, IMS does not serialize those databases before calling the SR.

Reentrant
: A single copy of the SR is used for the databases, and IMS does not serialize the database before calling the SR.

An SR for a PHDAM database is defined for a partition of the PHDAM database, and the SR is loaded when the partition is opened. Each PHDAM SR (for a partition) can have one of the following attributes:

Nonreusable
: IMS serializes the partition before calling the SR. If the SR is used for multiple partitions or databases, each partition or database has its own copy of the SR.

  This is the default attribute of the PHDAM SR generated by SRG.

Reusable
: IMS serializes the partition before calling the SR. If the SR is used for multiple partitions or databases, IMS does not serialize those partitions or databases before calling the SR.

Reentrant
: A single copy of the SR is used for the partitions and databases that use the SR. IMS does not serialize the partitions and databases before calling the SR.

For a general description of the randomizing routines for HDAM and PHDAM databases, see *IMS Exit Routines*.

A nonreusable HDAM or PHDAM SR does the following:

1. Compresses the input database keys.
2. Uses the compressed keys and calculates the relative block number and RAP number within the block.

A reusable or reentrant HDAM or PHDAM SR allocate a 2KB work area in ECSA for the IMS dependent region, if none has been allocated; the SR uses the work area to do the above calculations.

Abend

If an SR is abnormally ended while IMS is running, the dependent region that issued the DL/I call ends abnormally (DEDB), or the control region ends abnormally (HDAM and PHDAM).

SR action for an unexpected key

SR uses the SRT to determine the RAP number and, in the case of DEDB, the DMAC address of the area. When SR recognizes a key that cannot be randomized, SR sends a return code to IMS and displays a message on the OS console.

Unexpected database keys are:

- Key length specified by DBDGEN does not match the one specified during the SR generation.

- Key attribute of the input database key does not match the one specified during the SR generation (only in the case of DEDB SR).
- Assigned key is outside the range specified during the SR generation.

The processing for unexpected database keys differs depending on which option, CONT or ABEND, was specified for ERPROC parameter (of ANALYZE control statement) when the SR was generated.

HDAM and PHDAM

If the SR is passed a database key whose attribute does not match the one specified during the SR generation, then:

- If ERPROC=CONT is specified, the SR returns X'04' as the return code to IMS, and then IMS returns the status code "FM" to the application program that issued the DL/I call.
- If ERPROC=ABEND is specified, the SR returns X'08' as the return code to IMS, and then IMS abends the application program.

If the SR is passed a database key whose length does not match the one specified during the SR generation, then, regardless of the ERPROC parameter specification, the SR always returns the return code X'08' to IMS and the application program abends.

The following table describes the relationship between the return codes from the SR and the IMS processing. See "Messages and codes" on page 130 for the details of the messages displayed on the OS console by the SR.

Table 6. SR return codes and IMS processing (HDAM and PHDAM)

SR Return Codes	IMS Processing	Messages issued on the OS console by SR
0	Continues processing.	----
4 X'7FFFFF' → BLOCK# max RAP# → RAP#	Returns the status code "FM" to the application program.	FABO0002A
8	Application program ends abnormally with U812.	FABO0001A FABO0002A

DEDB

If the SR is passed a database key whose attribute does not match the one specified during the SR generation, then:

- If ERPROC=CONT is specified, the SR returns X'04' as the return code to IMS, and then IMS returns the status code 'FM' to the application program that issued the DL/I call.
- If ERPROC=ABEND is specified, the SR returns X'08' as the return code to IMS, and then IMS abends the application program.

If the SR is passed a database key whose length does not match the one specified during the SR generation, then, regardless of the ERPROC parameter specification, the SR always returns the return code X'08' to IMS and the application program abends.

The following table describes the relationship between the return codes issued by SR and the IMS processing. See “Messages and codes” on page 130 for the details of the messages displayed on the OS console by the SR.

Table 7. SR return codes and IMS processing (DEDB)

SR return codes	IMS processing	Messages issued on the OS console by SR
0	Continues processing.	----
4	Returns a status code 'FM' to the application program.	FABO0002A FABO0003A
8	Dependent region ends abnormally with U1021.	FABO0001A FABO0002A FABO0003A FABO0005A FABO0007A

Chapter 3. Examples

The following topics present examples of typical uses of SRG. The examples assume that the IBM-supplied cataloged procedure FABOSRG is being used.

Topics:

Example 1: Generating an SR for an HDAM database

This example represents the generation of SR for an HDAM database. In this example, option 1 is used for analyzing 200-byte database keys of the HDAM database.

The following figure shows the sample JCL for this job.

```
//SRGSR1A  JOB MSGLEVEL=(1,1)
//*------------------------------------------------------------------*
//* DATABASE IS ASSUMED TO BE CREATED ON DASD OF DEVICE TYPE 3390     *
//*------------------------------------------------------------------*
//SRG     EXEC FABOSRG,              * Use the cataloged procedure
//             INPTDSN='SRG.HDAMKEY', * Name of the key file
//             SERIN=SRGWRK,          * Volume serial for key file
//             INUNIT=3390            * Unit for key file
//GENERATE.SYSIN DD *
SRGSR1 ANALYZE (1,10,X),         // Hexadecimal field                *
               (11,20,A),        // Alphabetic field                 *
               (21,25,S),        // Signed packed decimal field      *
               (26,28,P),        // Unsigned packed decimal field    *
               (29,150,C),       // Alphanumeric field               *
               (151,200,A),      // Alphabetic field                 *
               TYPE=OSAM,        // HDAM DB with OSAM organization   *
               INKEYL=200,       // Length of a key                  *
               INLRECL=210,      // LRECL of key file                *
               INBLKSZ=2100,     // Block size of key file           *
               INOFFST=3,        // Start column of key in key file  *
               CIBYTE=4096,      // Block size of primary data set   *
               CITRK=12,         // # of blocks per TRK of 3390 DASD *
               TRCYL=15,         // # of TRKs per CYL of 3390 DASD   *
               PRIMCI=30,        // # of CYLs for RAA                *
               DBRECSZ=300,      // Average length of DB records     *
               OPTN=1,           // Key analysis option              *
               OPTMZ=10,         // Optimization option              *
               AREANM=SRGSR1T,   // Name of SRT                      *
               CORE=700,         // Initial size of equation table   *
               ERPROC=CONT,      // Error processing option          *
               PRINTNG=NO        // Report printing option
       END
/*
//
```

Figure 14. Example 1: Generating an SR for an HDAM database

Example 2: Regenerating the SR for an HDAM database

The following figure shows an example of SR regeneration (SRGEN=YES) for an HDAM database. SR regeneration is the assembling and link-editing of the SR source without making changes to the equation table.

```
//SRGSR1A  JOB MSGLEVEL=(1,1)
//*-----------------------------------------------------------------*
//* DATABASE IS ASSUMED TO BE CREATED ON DASD OF DEVICE TYPE 3390     *
//*-----------------------------------------------------------------*
//SRG     EXEC FABOSRG,              * Use the cataloged procedure
//             INPTDSN='SRG.HDAMKEY',  * Name of the key file
//             SERIN=SRGWRK,          * Volume serial for key file
//             INUNIT=3390            * Unit for key file
//GENERATE.SYSIN DD *
SRGSR1 ANALYZE (1,10,X),          // Hexadecimal field                *
               (11,20,A),         // Alphabetic field                 *
               (21,25,S),         // Signed packed decimal field      *
               (26,28,P),         // Unsigned packed decimal field    *
               (29,150,C),        // Alphanumeric field               *
               (151,200,A),       // Alphabetic field                 *
               TYPE=OSAM,         // HDAM DB with OSAM organization   *
               INKEYL=200,        // Length of a key                  *
               INLRECL=210,       // LRECL of key file                *
               INBLKSZ=2100,      // Block size of key file           *
               INOFFST=3,         // Start column of key in key file  *
               CIBYTE=4096,       // Block size of primary data set   *
               CITRK=12,          // # of blocks per TRK of 3390 DASD *
               TRCYL=15,          // # of TRKs per CYL of 3390 DASD   *
               PRIMCI=30,         // # of CYLs for RAA                *
               DBRECSZ=300,       // Average length of DB records     *
               OPTN=1,            // Key analysis option              *
               OPTMZ=10,          // Optimization option              *
               AREANM=SRGSR1T,    // Name of SRT                      *
               CORE=700,          // Initial size of equation table   *
               ERPROC=CONT,       // Error processing option          *
               PRINTNG=NO,        // Report printing option           *
               SRGEN=YES  <--- Generate SR from exisiting source files
       END
/*
//
```

Figure 15. Example 2: Regenerating the SR for an HDAM database

Example 3: Generating an SR for a DEDB database

This example shows one method of generating an SR for a DEDB database.

In this example, an input database key file is created in advance for each of the three DEDB areas, and they are analyzed separately as three jobs (step 1). The corresponding input database key file is specified for each job.

After running each job, the output reports should be checked for the synonym distribution (whether the number of RAPs does not exceed the maximum value) and other randomizer statistical information. Step 1 may be repeated several times with different parameters, if the result is unsatisfactory.

After SR source is created, the LINKSR job is run to generate an SR load module (step 2).

1. The following are the JCLs for analyzing database keys for each DEDB area using option 3.

```
//SRGSR2A  JOB MSGLEVEL=(1,1)
//SRG     EXEC FABOSRG,INPTDSN='NULLFILE'
//GENERATE.SRGSR21 DD DSN=SRG.KEYS2A11,DISP=OLD
//GENERATE.SYSIN   DD *
SRGSR2 ANALYZE (1,4,X),(5,8,S),(9,16,C),                              *
               INLRECL=32,INBLKSZ=6400,INOFFST=17,NOAREA=3,ERPROC=CONT,*
               TYPE=DEDB,OPTN=3,INKEYL=16,AREANM=SRGSR2T,OPTMZ=(1)
SRGSR21 AREADEF UOW=(70,20),ROOT=(100,20),TSIZE=2000,SIZE=8192,       *
               AVRECL=200,DELCHAR=-,CVCHAR=@,                          *
               KEYRNG=(00000000-0000000C-@@@@@@@@,0BE0FFFF-9999999C-@@@*
               @@@@@)
       END
/*
//
```

Figure 16. Example 3: Generating an SR for DEDB (Step 1) (Part 1 of 3)

```
//SRGSR2B  JOB MSGLEVEL=(1,1)
//SRG     EXEC FABOSRG,INPTDSN='NULLFILE'
//GENERATE.SRGSR22 DD DSN=SRG.KEYS2A21,DISP=OLD
//GENERATE.SYSIN DD *
SRGSR2 ANALYZE (1,4,X),(5,8,S),(9,16,C),                              *
               INLRECL=32,INBLKSZ=6400,INOFFST=17,NOAREA=3,ERPROC=CONT,*
               TYPE=DEDB,OPTN=3,INKEYL=16,AREANM=SRGSR2T,OPTMZ=(1)
SRGSR22 AREADEF UOW=(70,20),ROOT=(100,20),TSIZE=2000,SIZE=512,        *
               AVRECL=200,DELCHAR=-,CVCHAR=@,                          *
               KEYRNG=(0BE10000-0000000C-@@@@@@@@,1737FFFF-9999999C-@@@*
               @@@@@)
       END
/*
//
```

Figure 17. Example 3: Generating an SR for DEDB (Step 1) (Part 2 of 3)

```
//SRGSR2C  JOB MSGLEVEL=(1,1)
//SRG     EXEC FABOSRG,INPTDSN='NULLFILE'
//GENERATE.SRGSR23 DD DSN=SRG.KEYS2A30,DISP=OLD
//GENERATE.SYSIN DD *
SRGSR2 ANALYZE (1,4,X),(5,8,S),(9,16,C),                              *
               INLRECL=32,INBLKSZ=6400,INOFFST=17,NOAREA=3,ERPROC=CONT,*
               TYPE=DEDB,OPTN=3,INKEYL=16,AREANM=SRGSR2T,OPTMZ=(1)
SRGSR23 AREADEF UOW=(51,01),ROOT=(061,01),TSIZE=2000,SIZE=512,        *
               AVRECL=200,DELCHAR=-,CVCHAR=@,                          *
               KEYRNG=(17380000-0000000C-@@@@@@@@,1CEFFFFF-9999999C-@@@*
               @@@@@)
       END
/*
//
```

Figure 18. Example 3: Generating an SR for DEDB (Step 1) (Part 3 of 3)

2. The following figure shows the JCL for SR generation and registration using the SR source.

```
//SRGSR2L1 JOB MSGLEVEL=(1,1)
//SRG     EXEC FABOSRG,INPTDSN='NULLFILE'
//GENERATE.SYSIN DD *
SRGSR2 ANALYZE (1,4,X),(5,8,S),(9,16,C),                              *
               INLRECL=32,INBLKSZ=6400,INOFFST=17,NOAREA=3,ERPROC=CONT,*
               TYPE=DEDB,OPTN=3,INKEYL=16,AREANM=SRGSR2T,OPTMZ=(1)
       LINKSR AREAID=SRGSR21,SRGSR22,SRGSR23
       END
/*
//
```

Figure 19. Example 3: Generating an SR for DEDB (Step 2)

Example 4: Re-creating a DEDB area equation table

This example assumes that the DEDB SR generated in Example 3 is being used.

This example represents the re-creation of a DEDB area equation table. While using the generated SR for a DEDB database, synonyms may cluster in a specific DEDB area. In such a case, you should run SRG to re-create the equation table for the DEDB area.

Follow the steps described below:

1. Re-create the database key file of the target DEDB area.
2. Modify the SRG control statements that were used in Step 1 of Example 3 as follows:
   - Use the same parameters for the ANALYZE control statement.
   - Prepare an AREADEF control statement for the target DEDB area.

   The following figure shows the sample JCL.

```
//SRGSR2A  JOB MSGLEVEL=(1,1)
//SRG     EXEC FABOSRG,INPTDSN='NULLFILE'
//GENERATE.SRGSR21 DD DSN=SRG.KEYS2A11,DISP=OLD
//GENERATE.SYSIN   DD *
SRGSR2 ANALYZE (1,4,X),(5,8,S),(9,16,C),                               *
               INLRECL=32,INBLKSZ=6400,INOFFST=17,NOAREA=3,ERPROC=CONT,*
               TYPE=DEDB,OPTN=3,INKEYL=16,AREANM=SRGSR2T,OPTMZ=(1)
SRGSR21 AREADEF UOW=(70,20),ROOT=(100,20),TSIZE=2000,SIZE=8192,        *
               AVRECL=200,DELCHAR=-,CVCHAR=@,                          *
               KEYRNG=(00000000-0000000C-@@@@@@@@,0BE0FFFF-9999999C-@@@*
               @@@@@)
       END
/*
//
```

Figure 20. Example 4: Re-creating a DEDB area equation table (Step 2)

3. Run the SRG job.
4. If the result of the job is satisfactory, specify the LINKSR control statement and rerun the SRG job to generate a new SR load module to reflect the changes in SRT source. You must specify the target DEDB area for the LINKID parameter, and all the DEDB areas for the AREAID parameter.

   The following figure shows the sample JCL.

```
//SRGSR2L2 JOB MSGLEVEL=(1,1)
//SRG     EXEC FABOSRG,INPTDSN='NULLFILE'
//GENERATE.SYSIN DD *
SRGSR2 ANALYZE (1,4,X),(5,8,S),(9,16,C),                               *
               INLRECL=32,INBLKSZ=6400,INOFFST=17,NOAREA=3,ERPROC=CONT,*
               TYPE=DEDB,OPTN=3,INKEYL=16,AREANM=SRGSR2T,OPTMZ=(1)
       LINKSR  AREAID=SRGSR21,SRGSR22,SRGSR23,                         *
               LINKID=SRGSR21
       END
/*
//
```

Figure 21. Example 4: Re-creating a DEDB area equation table (Step 4)

Example 5: Adding a DEDB area equation table

This example assumes that the DEDB SR generated in Example 3 is being used.

This example represents the addition of a DEDB area equation table. While using the generated SR for a DEDB database, a new DEDB area may be added to the database. In such a case, you should run SRG to add the equation table for the DEDB area.

Follow the steps described below:

1. Create a database key file for the target DEDB area.
2. Modify the SRG control statements that were used in Step 1 of Example 3 as follows:
   - Use the same parameter for the ANALYZE control statement.
   - Prepare an AREADEF control statement for the target DEDB area.

   The following figure shows the sample JCL.

```
//SRGSR2D  JOB MSGLEVEL=(1,1)
//SRG     EXEC FABOSRG,INPTDSN='NULLFILE'
//GENERATE.SRGSR24 DD DSN=SRG.KEYS2A40,DISP=OLD
//GENERATE.SYSIN DD *
SRGSR2 ANALYZE (1,4,X),(5,8,S),(9,16,C),                               *
               INLRECL=32,INBLKSZ=6400,INOFFST=17,NOAREA=4,ERPROC=CONT,*
               TYPE=DEDB,OPTN=3,INKEYL=16,AREANM=SRGSR2T,OPTMZ=(1)
SRGSR24 AREADEF UOW=(51,01),ROOT=(061,01),TSIZE=2000,SIZE=512,         *
               AVRECL=200,DELCHAR=-,CVCHAR=@,                          *
               KEYRNG=(1CF00000-0000000C-@@@@@@@@,FFFFFFFF-9999999C-@@@*
               @@@@@)
      END
/*
//
```

Figure 22. Example 5: Adding a DEDB area equation table (Step 2)

3. Run the SRG job.
4. If the result of the job is satisfactory, specify the LINKSR control statement and rerun the SRG job to generate a new SR load module to reflect the changes in SRT source. You must specify the target DEDB area for the LINKID parameter, and all the DEDB areas for the AREAID parameter.

   The following figure shows the sample JCL.

```
//SRGSR2L3 JOB MSGLEVEL=(1,1)
//SRG     EXEC FABOSRG,INPTDSN='NULLFILE'
//GENERATE.SYSIN DD *
SRGSR2 ANALYZE (1,4,X),(5,8,S),(9,16,C),                               *
               INLRECL=32,INBLKSZ=6400,INOFFST=17,NOAREA=4,ERPROC=CONT,*
               TYPE=DEDB,OPTN=3,INKEYL=16,AREANM=SRGSR2T,OPTMZ=(1)
      LINKSR  AREAID=SRGSR21,SRGSR22,SRGSR23,SRGSR24,                  *
               LINKID=SRGSR24
      END
/*
//
```

Figure 23. Example 5: Adding a DEDB area equation table (Step 4)

Example 6: Generating an SR for a PHDAM partition

This example represents the generation of an SR for a partition PART1 of a PHDAM database. In this example, option 1 is used for analyzing 200-byte database keys of the partition PART1.

The following figure shows a sample JCL for this job. In this example, because ATTTRIB=RENT is specified, the SR is generated with reentrant attribute; also, because ERPROC=CONT is specified, the SR returns the status code FM when an invalid key is passed.

```
//SRGSR6  JOB MSGLEVEL=(1,1)
//*------------------------------------------------------------------*
//* PARTITION IS ASSUMED TO BE CREATED ON DASD OF DEVICE TYPE 3390    *
//*------------------------------------------------------------------*
//SRG    EXEC FABOSRG,INPTDSN='SRG.PART1KEY',
//            SERIN=SRGWRK,INUNIT=3390
//GENERATE.SYSIN DD *
PART1SR ANALYZE (1,10,X),(11,20,A),(21,25,S),(26,28,P),(29,150,C),     *
               (151,200,A),                                            *
               INLRECL=210,                                            *
               INBLKSZ=2100,                                           *
               INOFFST=3,                                              *
               CORE=700,                                               *
               TYPE=OSAM,                                              *
               OPTN=1,                                                 *
               OPTMZ=3,                                                *
               CIBYTE=4096,                                            *
               CITRK=12,                                               *
               TRCYL=15,                                               *
               PRIMCI=100,                                             *
               DBRECSZ=300,                                            *
               INKEYL=200,                                             *
               AREANM=PART1SRT,                                        *
               ERPROC=CONT,                                            *
               ATTRIB=RENT
        END
/*
//
```

Figure 24. Example 6: Generating an SR for a PHDAM partition

Chapter 4. Control statements

The following topics summarize the parameters for the ANALYZE, AREADEF, and LINKSR control statements.

Topics:

ANALYZE control statement

The following table summarizes the parameters for the ANALYZE control statement.

Table 8. ANALYZE control statement

Parameter	H for (P)HDAM D for DEDB	Required or optional	Contents	Default value
label	H, D	Required	SR Name	None
(a,b,c[,d])	H, D	Required	Key compression parameter Specify one of the following attributes for the parameter c: **A** Alphabetic **C** Alphanumeric **I** Ignored field **N** Numeric **O** Nonsequential randomizing module **P** Unsigned packed decimal **Q** Character **S** Signed packed decimal **U** Specific character **X** Hexadecimal **Y** Katakana **Z** Katakana and character	None
AREANM	H, D	Optional	SRT name	xxxxx##T
ATTRIB	H	Optional	Attribute of an HDAM or PDAM SR	blank
CALLTYPE	D	Optional	Type of the randomizer call interface	STD
CIBYTE	H	Optional	CI or block size of database	4096
CITRK	H	Optional	Number of CIs or blocks per track	3
CORE	H	Required	Size of the equation table	None
DBLEN	D (See Note 1)	(Optional)	Position of the database record length field	None
DBLOFFST	H (See Note 1)	(Optional)	Position of the database record length field	None
DBM	D	Optional	Has no meaning (this parameter is provided for the compatibility with DBT Version 2 SDO.)	Not used
DBRECSZ	H (See Note 1)	(Required)	Average database record length	None
ERPROC	H, D	Optional	Use of status code 'FM'	ABEND
FRECI	H	Optional	Number of unused CIs or blocks	1
INBLKSZ	H, D	Required	Block size of database key file	None
INKEYL	H, D	Required	Length of key field	None
INLRECL	H, D	Required	Length of key file record	None
INOFFST	H, D	Required	Start column of key field	None
KEYNBR	H, D	Optional	Creation of compressed key file on storage	(Created on DASD)
LNKPGM	H, D	Optional	Linker program to be used	LKED
NOAREA	D	Optional	Number of DEDB areas	None
OPTMTIME	H	Optional	Maximum time for equation table optimization	0
OPTMZ	H, D	Optional	Equation table optimizing parameters	(9999,100)
OPTN	H, D	Optional	Key analysis method (option processing)	3

Table 8. ANALYZE control statement (continued)

Parameter	H for (P)HDAM D for DEDB	Required or optional	Contents	Default value
PRIMCI	H	Optional	Size (number of cylinders) of the database	20
PRINTNG	H, D	Optional	Output reports/listings printing option	YES
RMODE	H, D	Optional	Parameter for compatibility with IMS/VS Aid Program Package SDO	None
SIZECHK	H, D	Optional	Parameter for issuing an error message when SR module size exceeds 16 MB	NO
SRCMIG	D	Optional	Parameter for migration	NO
SRGEN	H	Optional	Regeneration of SR for the HDAM database	NO
SRTYPE	D	Optional	SR load module type	SINGLE
TLIMIT	H	Optional	Maximum SRT size during equation table optimization	9999999
TNUMLIM	H	Optional	Maximum number of equation tables for equation table optimization	0
TRCYL	H	Optional	Number of tracks per cylinder	19
TYPE	H, D	Optional	Database type	DEDB
TYPECHG	D	Optional	Parameter for change of SR load module type	NO

Notes:

1. DBLEN, DBLOFFST, DBRECSZ, and AVRECL parameters are interrelated.
2. Use DBLOFFST for HDAM and PHDAM partition. DBLEN is supported only the compatibility with the existing JCL streams.

AREADEF control statement

The following table summarizes the parameters for the AREADEF control statement.

Table 9. AREADEF control statement

Parameter	HDAM(H) DEDB(D)	Required or optional	Contents	Default value
label	D	Required	DEDB area name	None
ARETRY	D	(See Note 2)	(See Note 2)	None
AREXT	D	Optional	Number of combinations for subkey ranges for the KEYRNG or the ARETRY parameter	1
AVRECL	D (See Note 1)	(Required)	Average database record length	None
CVCHAR	D	Optional	Space replacement character for the KEYRNG or the ARETRY parameter	None
DELCHAR	D	Optional	Delimiter character for the KEYRNG or the ARETRY parameter	None
KEYRNG	D	Required	Key range of the area	None
OPTMTIME	D	Optional	Maximum time for equation table optimization	0
OPTMZ	D	Optional	Equation table optimizing parameters	(9999,100)
ROOT	D	Required	Same as DBDGEN ROOT parameter	None
SIZE	D	Required	CI size of DEDB	None
TLIMIT	D	Optional	Maximum SRT size during equation table optimization	9999999
TNUMLIM	D	Optional	Maximum number of equation tables for equation table optimization	0
TSIZE	D	Required	Size of equation table	None
UOW	D	Required	Same as DBDGEN UOW parameter	None

Notes:

1. DBLEN, DBRECSZ, and AVRECL parameters are interrelated.
2. For information about the ARETRY parameter, see the explanation of the AREADEF control statement in "AREADEF control statement" on page 48.

LINKSR control statement

The following table summarizes the parameters for the LINKSR control statement.

Table 10. LINKSR control statement

Parameter	H for (P)HDAM D for DEDB	Required or optional	Contents	Default value
AREAID	D	Required	All the area names in DEDB	None
LINKID	D	Optional	Areas to be assembled and link-edited	ALL
SRCCHK	D	Optional	Level of the SR source file checking that should be done during the LINKSR process	0

Chapter 5. Reference: How the Sequential Randomizer is generated

The following topics describe in detail how the Sequential Randomizer (SR) is generated by SRG and how the generated SR works. The nonsequential randomizing module is generated in the same way.

Topics:

Compressing database keys

After analyzing the control statements specified by the user, SRG generates a database key compression routine and compresses the input database keys in order to improve the key analysis efficiency. Key compression is performed according to the key attributes specified in the control statement.

A single key attribute can be specified for an entire field, or different key attributes can be specified for each subkey field. The compressed keys are stored in a work data set called *compressed key file*, or in private storage above 16 MB line (see the KEYNBR parameter description in KEYNBR).

The generated key compression routine becomes a part of the SR and is used whenever SR is called during IMS execution.

The following table shows the subkey field attributes and the compressed key length:

Table 11. Key Attributes and Compressed Key Length

Subkey Field Attribute	Parameter Value	Original Key Length and Compressed Key Length
Hexadecimal data	X	8 bits → 8 bits
Unsigned packed decimal data	P	1 byte → 7 bits 2 bytes → 14 bits 3 bytes → 20 bits 4 bytes → 27 bits
Signed packed decimal data	S	1 byte → 4 bits 2 bytes → 10 bits 3 bytes → 17 bits 4 bytes → 24 bits 5 bytes → 30 bits
Alphabetic data (including a blank)	A	8 bits → 5 bits
Numeric data (0 to 9)	N	1 byte → 4 bits 2 bytes → 7 bits 3 bytes → 10 bits 4 bytes → 14 bits 5 bytes → 17 bits 6 bytes → 20 bits 7 bytes → 24 bits 8 bytes → 27 bits 9 bytes → 30 bits
Alphanumeric data (including a blank)	C	8 bits → 6 bits
Character data	Q	8 bits → 6 bits
Specific character data	U	Depends on parameter
Katakana*	Y	8 bits → 6 bits
Katakana and character data	Z	8 bits → 7 bits
Ignored data	I	1 byte (fixed)
*: Katakana is a Japanese unique phonetic character set.		

Analyzing database keys and creating the equation table

If the number of RAPs is equal to the number of input database keys, database keys can be mapped one-to-one to the RAP numbers, and the physical sequence of the root segments can be made to match the logical sequence. However, this requires a one-to-one database key to the RAP number mapping table, which is difficult to obtain when the storage size is limited. Therefore, SRG uses linear equations to map multiple database keys to multiple RAP numbers.

SRG analyzes the compressed key values, divides the keys into groups, and generates a linear equation for each group that associates the compressed key values within the group with *relative* RAP numbers.

Note: SRG internally assigns a sequence number (relative RAP number) to each RAP, throughout a database for HDAM, or throughout a partition for PHDAM.

The generated linear equations are stored in a table called the *equation table*, which is a part of the generated SR. The equation table contains key range values and the associated linear equations, and is referred to by SRR.

The following figure illustrates the equation table and shows how database keys are associated with relative RAP numbers:

Equation table

Range of keys	Linear equation
K_1 -	$y = a_1 x + b_1$
K_2 -	$y = a_2 x + b_2$
: :	: :
K_{n-1} -	$y = a_{n-1} x + b_{n-1}$
K_n -	$y = a_n x + b_n$

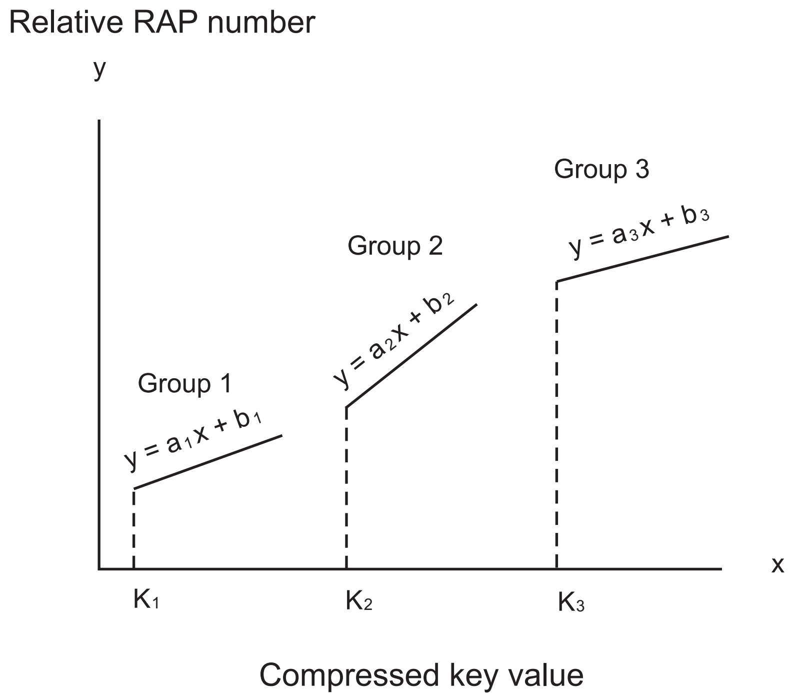


Figure 25. Database key groups and equation table

Upon grouping the database keys, SRG analyzes the keys to determine where the relative change in value (compressed key value) is the greatest and groups them at this point. The number of RAPs for the entire database is divided proportionally according to the number of keys in a group. Then a linear equation is created for each group.

For example, consider the following case:

1. Key values in the database key file..... 2, 3, 7, 9, 18, 20, 21, 27, 28
2. Number of key groups (number of equations) specified by the user 2
3. Number of RAPs in the entire database specified by the user 9

SRG analyzes the difference between consecutive keys or the change in the high-order bytes (or bits) to group the keys.

In this case, the key values 2 through 9 are grouped in one group and values 18 through 28 are grouped into another group, and linear equations (A and B) are created to link the minimum and maximum key values within each group. The number of RAPs assigned to each group is determined by dividing the number of RAPs of the entire database in proportion to the number of keys in each group.

The following relative RAP numbers are determined from these two equations:

Key Value	Equation	Relative RAP No.
2	A	1
3		1
7		3
9		4
18	B	5
20		6
21		6
27		9
28		9

Note: The results obtained from the equations are rounded because the relative RAP number must be an integer.

SRG calculates the number of equations to be created from the size of the equation table specified by the control statement. Then it analyzes the keys to create as many key groups as there are equations.

When a limited number of groups are used for key analysis, the processing time and the result depend on the database keys. Therefore, SRG provides three different analysis methods (options 0, 1, and 3) that can be selected by the user.

The analysis method (option) can be specified in the OPTN parameter of the ANALYZE statement. Option 0 analyzes the difference between consecutive keys. Option 1 analyzes the difference in the compressed key bytes starting from the high-order byte. Option 3 analyzes the difference in compressed key bits starting from the high-order bit.

See Chapter 7, “Reference: Database key analysis methods,” on page 111 for a detailed description of each method.

Optimizing the equation table

After the grouping of the database keys, if the key values are not evenly scattered within a group, synonyms may occur.

In the example in "Analyzing database keys and creating the equation table" on page 97, the same relative RAP numbers are assigned to keys 2 and 3, 20 and 21, and 27 and 28. SRG will adjust the equation if optimization is specified by the user.

If the control statement specifies not to generate synonyms, equations A and B will be divided into A1, A2 and B1, B2 respectively, and the relative RAP numbers are determined as shown in the following table:

Key Value	Equation	Relative RAP No.
2	A1	1
3		2
7	A2	3
9		4
18	B1	5
20		6
21		7
27	B2	8
28		9

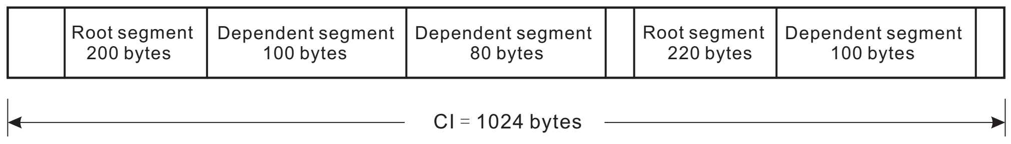
Conditions for optimization

The equation table created by the key analysis relates each input database key to a relative RAP number of the database. The creation of an equation table is based on the size of the table and the number of RAPs specified in the SRG control statement.

It takes no consideration of:
- The number of synonyms
- The percentage of database records in a CI

Note: The number of synonyms is the number of root segments with the same relative RAP number, excluding the first root segment pointed to by the RAP. For example, if the number of synonyms is zero, only one root segment is assigned to a RAP.

SRG enables users to optimize the equation table by specifying the following in the OPTMZ parameter:
- The maximum number of synonyms per RAP
- The maximum CI capacity ratio (the maximum percentage ofthe total database record length per CI or per block).



CI capacity ratio =((200+100+80)+(220+100))/1024 × 100 = 68%

Either the maximum number of synonyms or the maximum CI capacity ratio (,or both) can be specified. However, the treatment of these optimizing conditions for DEDB differs from their treatment for HDAM and PHDAM, as follows:

HDAM and PHDAM
Only the maximum number of synonyms is considered for optimization. The maximum CI capacity ratio is used only for determining the number of RAPs per CI when the SR is first created. In this way, you can obtain free space in a CI.

```
Average Record Length
  = ∑ (Database Record Length)/Total Number of Database Keys

Number of RAPs per CI
  = (CI size × Maximum CI Capacity Ratio)/Average Record Length

Total Number of RAPs = (Number of RAPs per CI) × Total Number of CIs
```

Figure 26. Optimizing Equation Table (HDAM and PHDAM)

Example:

Suppose the average record length of an HDAM database is 300 and the CI size is 1024 bytes. If the maximum CI capacity ratio is not specified, SRG calculates the number of RAPs per CI as follows:

```
(Number of RAPs per CI) = 1024 × 1.0  / 300
                        = 3.41
                        ≅ 3
```

If the maximum CI capacity ratio of 80 (%) is specified, the number of RAPs per CI is calculated as follows:

```
(Number of RAPs per CI) = 1024 × 0.8  / 300
                        = 2.73
                        ≅ 2
```

DEDB Both the maximum number of synonyms and the maximum CI capacity ratio are used as the optimization conditions.

Method of optimization

SRG creates a temporary SR load module and simulates the invoking of SR by IMS based on the input database keys to check if the optimization conditions are satisfied. If not, SRG adjusts the equation table by dividing the equations that do not satisfy the optimization conditions.

This process is repeated until the optimization conditions are satisfied. This optimization will generate an SR with fewer synonyms and better I/O efficiency, but the size of the equation table may be greater than that specified in the control statement, and the total number of RAPs may increase. The relationship between optimization conditions and the equation division is summarized in Table 12 and Table 13.

HDAM and PHDAM

Table 12. Relationship between optimization conditions and equation division (HDAM and PHDAM)

	Maximum number of synonyms is specified	**Maximum number of synonyms is *not* specified**
CI capacity ratio Is Specified or *not* specified	Equations are divided when the maximum number of synonyms is not satisfied.	Equations are *not* divided.

DEDB

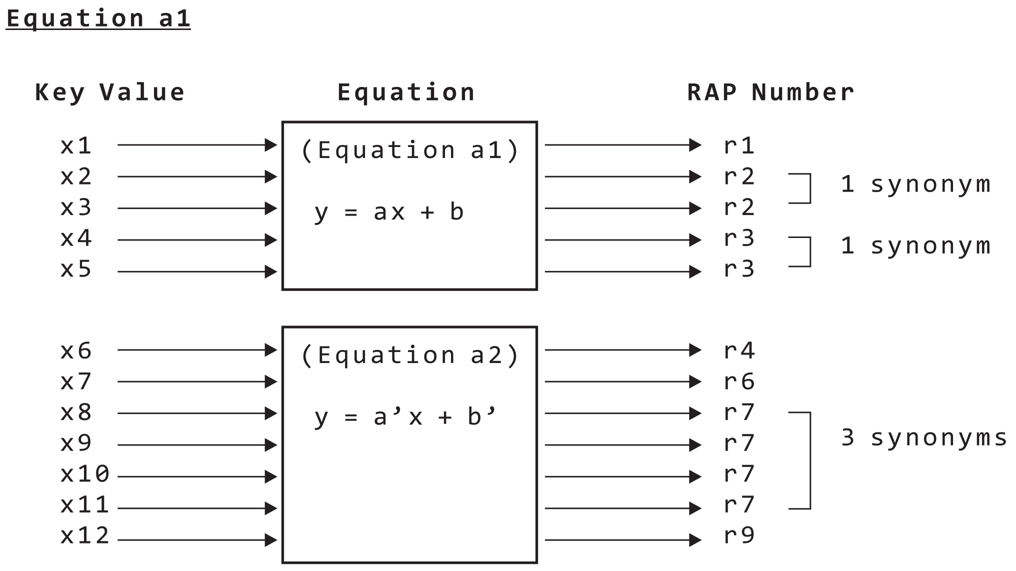
Table 13. Relationship between optimization conditions and equation division (DEDB)

	Maximum number of synonyms is specified	**Maximum number of synonyms is *not* specified**
CI capacity ratio is specified	Equations are divided when either of the two conditions is not satisfied.	Equations are divided when the CI capacity ratio is not satisfied.
CI capacity ratio is *not* specified	Equations are divided when the maximum number of synonyms is not satisfied.	Equations are ***not*** divided.

Note: When default values are specified for these conditions, they are treated as if the options were not specified.

Example

Suppose the specified optimization condition for the maximum number of synonyms is 2 when the RAP total is 100 and the key total is 100, and the equations are created as follows:



Equation a1 is not divided because the number of keys assigned to each RAP does not exceed the maximum number of synonyms. In equation a2, the number of synonyms for the RAP number 7 is 3, which exceeds the maximum number of synonyms; therefore, a new equation is created for the first three keys assigned to the RAP number 7.

Limiting the equation table size

When the OPTMZ parameter is specified in the SRG control statement, SRG optimizes the equation table by increasing the size of the table. By specifying the TLIMIT parameter or the TNUMLIM parameter of the SRG control statement (on the AREADEF statement for DEDB, or on the ANALYZE statement for HDAM and PHDAM), you can specify a maximum table size.

When TLIMIT is specified, SRG stops the optimization of an equation table when its size becomes equal to or greater than the specified value, and the equation table is generated. In this case, the conditions specified in the OPTMZ parameter may not be satisfied.

When TNUMLIM is specified, SRG stops the optimization of an equation table when the number of equation table entries becomes equal to or greater than the specified value, and generates the equation table. In this case, the conditions that are specified in the OPTMZ parameter might not be satisfied.

Limiting the optimization elapsed time

By specifying the OPTMTIME parameter on the AREADEF control statement (for DEDBs) or on the ANALYZE control statement (for HDAM and PHDAM databases), you can set a time limit for the equation table optimization processing.

If you specify the OPTMTIME parameter, SRG stops the optimization of the equation table when the elapsed time reaches the time limit and generates the equation table. In this case, the conditions that are specified in the OPTMZ parameter might not be satisfied.

Storing the SR sources

After the optimization of the equation table is completed (in case the optimization is specified), SRG once again simulates the invoking of the generated SR from IMS, based on the input database keys. Using the data obtained from simulation, SRG produces statistical information (such as the number of synonyms in the database and the CI capacity ratio) of the SR (for HDAM and PHDAM) or of each of the area units (for DEDB).

Then, SRG stores the SRR source (program part of SR source) and SRT source (equation tables) in the SR source library (SRRFILE).

In DEDB, an equation table must be created for each DEDB area in the database and stored in the SR source library before SR load module generation. In other words, the DEDB area processing (specified by ANALYZE control statement and AREADEF control statement) must be performed for all the areas. The SRR source, which is created by the first DEDB area processing, is updated every time a DEDB area is added or updated.

Generating an SR load module

Finally, the SR source (SRR source and SRT source) is assembled and link-edited to generate an SR load module, which is stored in the IMS load module library (SDFSRESL) or a user-specified library.

SRs are link-edited with AMODE=31 and RMODE=ANY.

The attribute of the SR is as follows:

HDAM and PHDAM

You can specify the reusability attribute for an HDAM or a PHDAM SR. To do so, use the ATTRIB parameter of the ANALYZE control statement. For the details, see the description of the ATTRIB parameter. See also the section "How the SR works for an HDAM or PHDAM database" on page 76 which explains the characteristics of HDAM and PHDAM SR of each attribute.

DEDB When DEDB SR is a single-module type, SR becomes a reenterable module. When DEDB SR is a multiple-module type, the SRR module and each area equation table become reenterable modules, but the extended master table is link-edited with the REUS attribute, because the addresses of area equation tables are set in the table at runtime.

SRT (Sequential Randomizing Table)

SR is composed of two parts: Sequential Randomizing Routine (SRR) and Sequential Randomizing Table (SRT). SRT is a table or a set of tables used by SRR for relating input database keys with RAP numbers.

In DEDB, SRT consists of multiple equation tables (one table for each DEDB area) and a master table that indicates the area range. If a DEDB SR is created by specifying SRTYPE=SINGLE, the SRT is linked with SRR and the SR is generated as a single module. If a DEDB SR is created by specifying SRTYPE=MULTI, the SRT is generated as multiple modules which consist of the extended master table module and the area equation table modules (one area equation table module for one area). The extended master table is an extension of the usual master table. In HDAM and PHDAM, SRT consists of only one equation table.

The following figure shows the difference in structure between DEDB SRT and HDAM/PHDAM SRT.

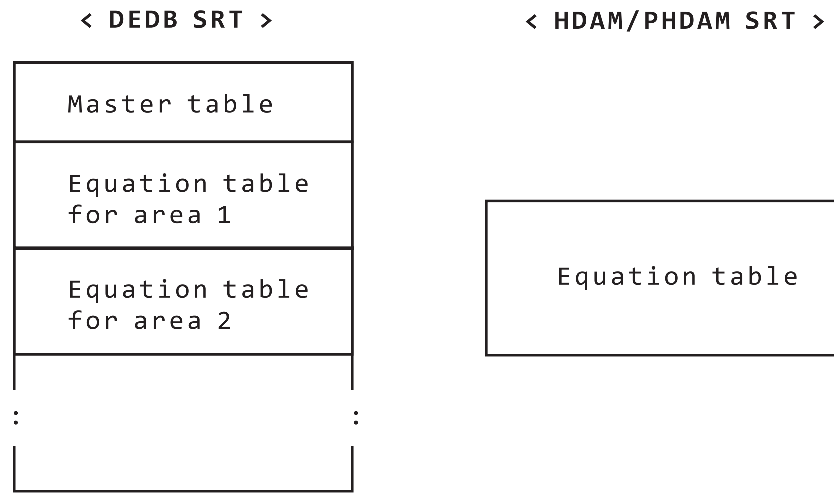
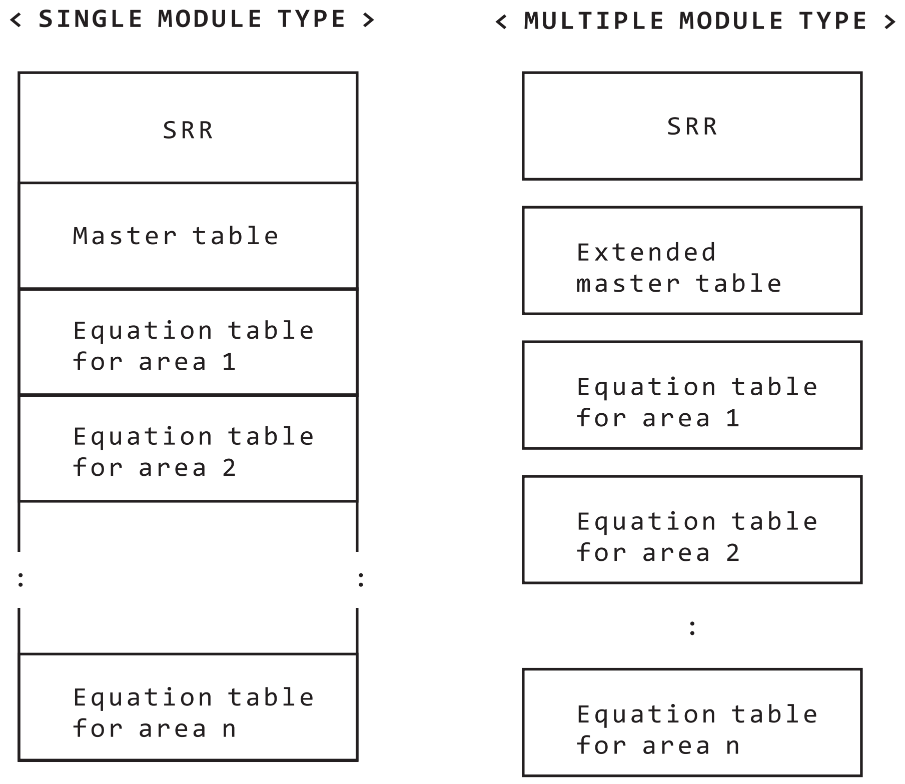


Figure 27. DEDB SRT and HDAM/PHDAM SRT

The following figure shows the differences in the load module structure of DEDB SR between single module type and multiple module type.



< SINGLE MODULE TYPE >
SRR
Master table
Equation table for area 1
Equation table for area 2
Equation table for area n
< MULTIPLE MODULE TYPE >
SRR
Extended master table
Equation table for area 1
Equation table for area 2
Equation table for area n

Figure 28. Structure of DEDB SR

Chapter 6. Reference: Required amount of resources

The following topics describe the resources that are required to run SRG and how to estimate the amount of each resource.

Topics:

Required resources

Equation table size, work data set size, and work storage size must be estimated.

- "Equation table size"
- "Work data set size "
- "Work storage size"

Equation table size

The equation table size should be specified by CORE and TSIZE parameters after considering the following factors:

- Number of keys to be entered
- Number of RAPs to be used

  HDAM and PHDAM (PRIMCI, CIBYTE, CITRK, TRCYL) DEDB (ROOT, UOW)
- Length of input key (INKEYL)
- Attributes of input key ((a,b,c) or (a,b,c,d))
- Equation Table generation method (OPTN)

Work data set size

The sizes of the following work data sets are greatly affected by the above parameters and the equation table size:

- Compressed key file (ddname: KEY)

  The size of the compressed key file is calculated as follows:

  {(Key length) + (Compressed key length) + 2 } × (Number of keys)
- Intermediate equation table data sets (ddnames: EQF, OUTS, EQT, and WORK).

  The sizes of the above data sets are defined by the values specified with IP= and JP= parameters of the cataloged procedure FABOSRG.
- Sort work data sets (ddname: SORTWK01-05)

  The sizes of sort work data sets are defined by the value specified with PRMSRT= parameter of the cataloged procedure FABOSRG.

The size of the work data set with ddname LKEDOUT might have to be changed when SRTYPE=MULTI is specified.

- If the number of AREADEF statements for the SRG job is, say, NUMAREA, then SRG requires

  NUMAREA + 3

  directory entries for this data set. So specify the size which can admit this number of directory entries. The size of this data set is defined by the values specified with LOSPC=, LOP=, LOS= and LOD= parameters of the cataloged procedure FABOSRG.

The size of the work data set with ddname LKEDIN might have to be changed when TYPECHG=YES is specified with SRTYPE=MULTI, since the type change to multiple modules needs more space for LKEDIN than the usual processing.

- The size of this data set is defined by the values specified with LSPC=, LP= and LS= parameters of the cataloged procedure FABOSRG.

Work storage size

Temporary SR size

During the SR generation, SRG loads the generated temporary SR into the extended private area in order to simulate the invoking of the SR by IMS. The size of the temporary SR is calculated as follows:

(Temporary SR size) = (SRR size) + (Equation table size)

The approximate SRR size is in the range of 3.5 K to 13 K bytes (HDAM and PHDAM), or 2.5 K to 12 K bytes (DEDB). The size increases in accordance with the number of subkey fields specified by the compression parameter ((a,b,c) or (a,b,c,d)) of the ANALYZE control statement.

Compressed key file on storage

The KEYNBR parameter in the ANALYZE control statement specifies that the compressed key file is to be created in storage above 16 MB line. The size of the compressed key file is calculated as follows:

{(Key length) + (Compressed key length) + 2} ×
(KEYNBR parameter value)

Work storage for KEYRNG or ARETRY parameter on AREADEF control statement

The work storage for ARETRY is obtained in the storage above 16 MB line. The work storage size is calculated as follows:

Let #CHAR be the total number of characters in the key specification of a KEYRNG or ARETRY parameter, excluding the character that is specified by DELCHAR and including CVCHAR. For example, if an AREADEF control statement is specified as:

```
AREANAME  AREADEF                                                *
                UOW=...                                          *
                KEYRNG=                                          *
                (00000000-0000001C-@@@@@@@@                      *
                ,0000C34F-1000000C-@@@@@@@@),                    *
                (0000C350-0000001C-@@@@@@@@                      *
                ,0001869F-1000000C-@@@@@@@@),                    *
                           .........
                DELCHAR=-,CVCHAR=@
```

then #CHAR is equal to 24. Let N be the integer that rounds up the value of #CHAR/80. In the preceding example, N is equal to 1. Let

A=(Number of key ranges) × (Key length) × N × 80 × 2
B=(Number of key ranges) × (Key length) × 2 + 8

then, the required storage size is (A + B) bytes.

The number of key ranges is the number of combinations of (*a*,*b*) in the KEYRNG or ARETRY parameter on the AREADEF control statement.

How to estimate the size of resources

This topic describes how to estimate the sizes of equation tables and work data sets from the number of input keys and the estimated number of equations.

We assume that

```
KEYL  = length, in bytes, of an input key
CKEYL = length, in bytes, of a compressed key
#EQU  = estimated number of equations
#KEY  = number of input keys
CAPA  = bytes in a cylinder, which is determined by
        number of the type of the DASD model and the block size
   c  = Key type coefficient*
```

Here, the coefficient c is determined as follows:

X (hexadecimal)	1
Y (Katakana)	0.9
P (unsigned Packed Decimal)	0.5
S (signed Packed Decimal)	0.5
C (alphanumeric)	0.8
N (Numeric)	0.4
I (ignored Key Field)	0.1
A (alphabetic)	0.8
Q (character)	0.9
O (nonsequential Randomizing Module)	0.2
Z (Katakana and Character)	0.9
U (specific Character)	0.1

When the above key types are mixed, the coefficient is determined by the following equation:

```
c =  [total sum of (length of each key type × key coefficient)]/
                           [total sum of (length of each key type)]
```

Then, we can estimate the size of the resources as follows:

```
CKEYL = KEYL × c

TSIZE (or CORE) (bytes) = #EQU × (CKEYL + 9)

KEY File (cylinders) = (#KEY × KEYL × c × 2) / (CAPA × 0.95)

Intermediate equation table data set or
sort work data set (OPTN=1 or 3) (cylinders)
              = (#EQU × KEYL × c × 2) / (CAPA × 0.95)

Intermediate equation table data set or
sort work data set (OPTN = 0) (cylinders)
              = KEY File (cylinders) × 3.5
```

Example

A key consists of two hexadecimal subkey fields: branch-number field (3 bytes) and account-number field (6 bytes). The compressed key length is 9 bytes. The keys with the branch number 001 to 010 are to be analyzed, and keys are grouped by the branch number. The equation table size is calculated as follows:

TSIZE (or CORE) $= 10 \times (9 + 9) = 180$

Chapter 7. Reference: Database key analysis methods

SRG provides three methods for analyzing database keys. Use the information in the following topics to determine the best option to use.

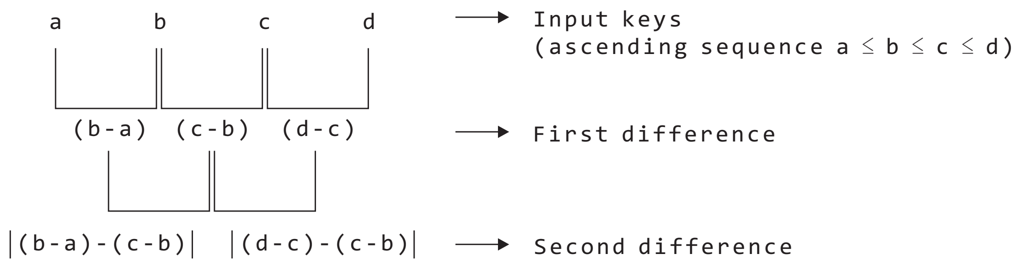
Topics:

- "Option 0" on page 112
- "Option 1" on page 116
- "Option 3" on page 117
- "Key analysis methods summary" on page 121

Option 0

With this method, the second differences of all keys are obtained, and the first key of each group is selected sequentially starting from the largest second difference.

The second differences are obtained as follows:



Advantages

Grouping is performed where the change in key is the greatest, thus preventing the clustering of synonyms.

If the number of groups can be estimated, an optimum table can be created by specifying the size of the equation table that is calculated from this number of groups, in the ANALYZE control statement.

If the differences between keys are the same (all first differences are all equal and the second differences are all 0), one equation table entry can be used to map the keys to RAP numbers.

An example showing the advantages of option 0 follows:

Example: Number of Keys = 12; Number of RAPs = 12; Divide into three groups

```
2   5    8    11    85    88    91    94    171    174    177    180   ——→ Input keys
 \ / \  /  \ /  \  /  \  /  \  /  \  /  \  /    \ /   \ /    \  /
  3   3     3    74    3     3     3     77      3     3      3       ——→ First difference
   \ /  \ /   \ /  \  /  \  /  \  /  \  /   \  /   \ /   \  /
    0    0    71    71    0      0    74     74     0      0          ——→ Second difference

   (5)  (6)   (3)   (4)  (7)    (8)   (1)    (2)   (9)    (10)
```

The first key of a group is determined in the following manner. The keys with the largest second difference are selected and the first differences of the three keys are compared to determine the first key. For example, if the three consecutive keys are LOW, MIDDLE, and HIGH:

when MIDDLE - LOW ≥ HIGH - MIDDLE, the first key is MIDDLE

when MIDDLE - LOW < HIGH - MIDDLE, the first key is HIGH.

In the above example, 171 is the first key for 1. The first key for 2 is also 171, but 2 is ignored in such cases and the first key is obtained from 3.

In this manner, the above example is grouped as follows:

(2, 5, 8, 11), (85, 88, 91, 94), (171, 174, 177, 180)

Four RAPs are assigned to each group because RAPs are divided in proportion to the number of keys in each group. This determines the slopes and offsets of the linear equations. As shown in the following figure, keys and RAPs are in one-to-one relationship, thus creating no synonyms.

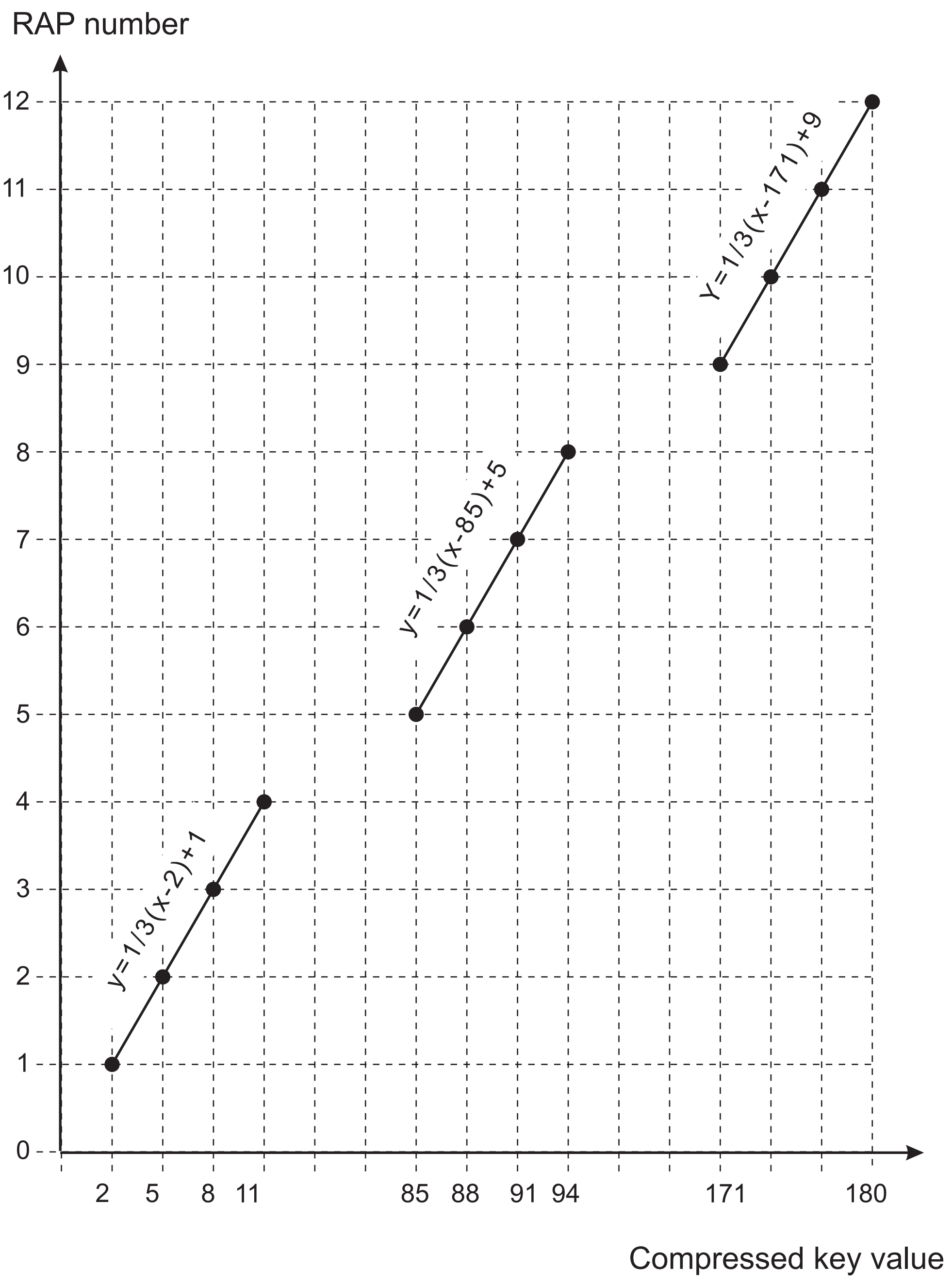


Figure 29. Option 0 - Example 1

Disadvantages

When the keys are not easily grouped because they are unevenly distributed, there is a greater chance of synonyms occurring.

An example showing the disadvantages of option 0 follows:

Example: Number of Keys = 12; Number of RAPs = 12; Divide into three groups

```
2   11   29   37    54    82   119   147    184    230    285    331
 \ /  \ /  \ /  \ /   \ /   \ /   \/    \ /    \ /    \ /    \  /
  9   18    8    17    28    37    28    37     46     55     46
   \ /  \  / \  /  \  /  \  /  \  /  \  /  \  /   \   /  \   /
    9    10    9    11    9     9     9      9      9      9

   (3)  (2)   (4)   (1)  (5)    (6)   (7)    (8)   (9)    (10)
```

The keys in this example are grouped as follows:

(2,11), (29,37,54), (82,119,147,184,230,285,331)

The three groups are assigned 2, 3, and 7 RAPs respectively, which is in proportion to the number of keys in each group.

From this, the slope and offset of the linear equation are determined. The relationship between keys and RAPs is shown in the following figure.

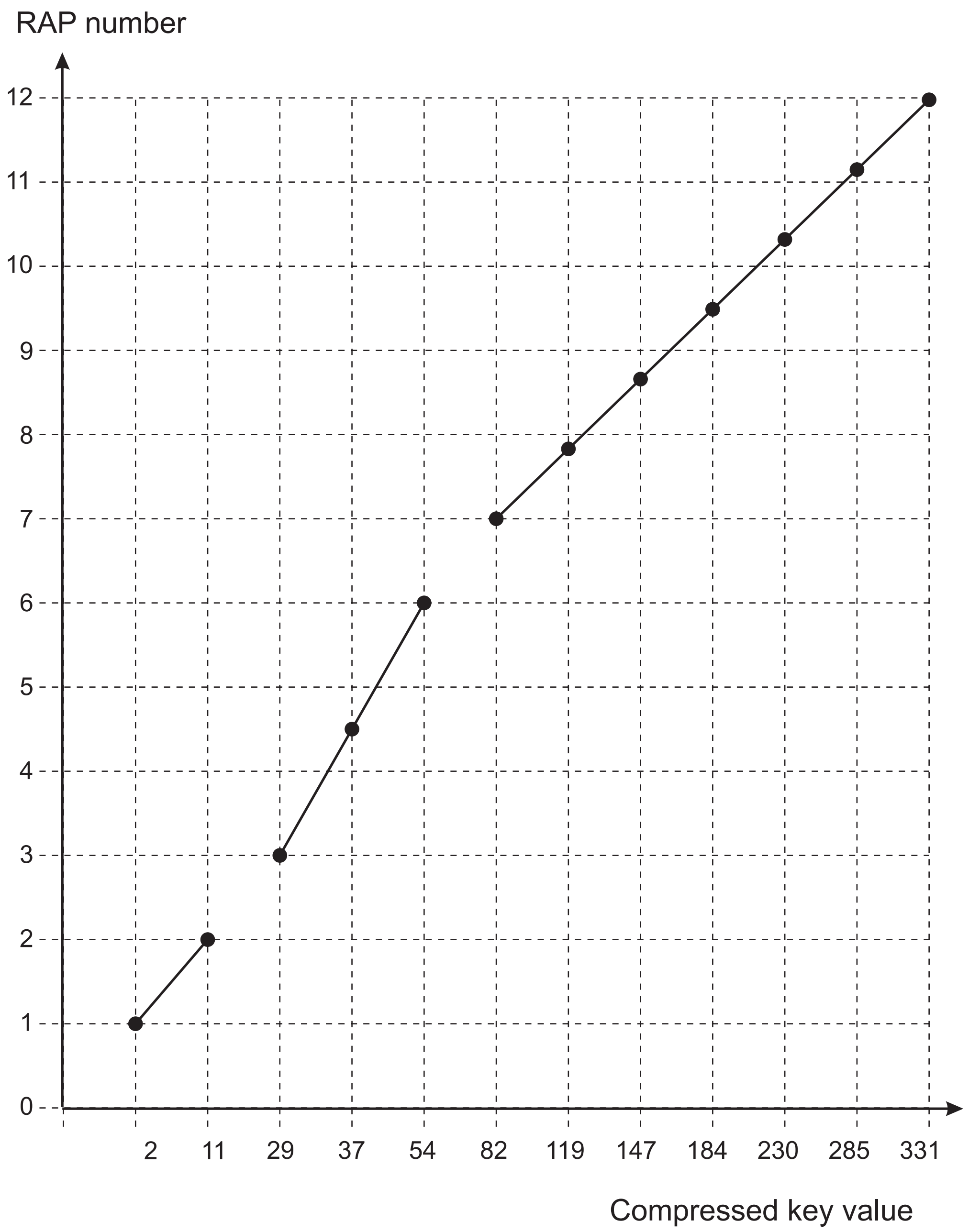


Figure 30. Option 0 - Example 2

Option 1

With this method, the compressed key values are analyzed starting from the high-order bytes to group the compressed database keys.

This method groups keys in large groups and takes into consideration their scattering.

Example:

```
000300
000301   Group  1
  :
000388
200400
200401   Group  2
  :
200497
500300
500301   Group  3
  :
500303
```

The bytes are analyzed from left to right until the number of groups satisfies the number of equations.

Advantages
: This method can be used for the database key files whose key sizes and distributions cannot be analyzed by option 0.

Disadvantages
: If the changes in compressed keys are in low-order bytes, the compressed keys may be analyzed many times, thus delaying SR generation. If the size of the defined equation table is small and thus the number of equations is satisfied while analyzing the first byte, all the remaining keys are grouped in the same group, which results in causing synonyms.

Option 3

With this method, the changes in compressed key bits are examined and keys are grouped where the change in bits is great.

If this method is selected, the changes in all compressed key bits are examined first to create a bit-pattern table.

Creating a bit-pattern table

The bit-pattern table contains the location of the first change bit of the compressed key for all compressed keys. The following figure is an example of a bit-pattern table generated for an eight-bit compressed key value.

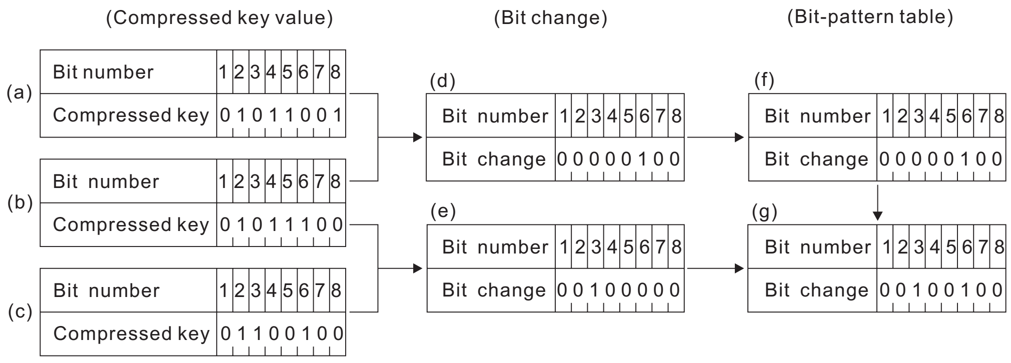

(Compressed key value)
(Bit change)
(Bit-pattern table)
(a)
Bit number 1 2 3 4 5 6 7 8
Compressed key 0 1 0 1 1 0 0 1
(b)
Bit number 1 2 3 4 5 6 7 8
Compressed key 0 1 0 1 1 1 0 0
(c)
Bit number 1 2 3 4 5 6 7 8
Compressed key 0 1 1 0 0 1 0 0
(d)
Bit number 1 2 3 4 5 6 7 8
Bit change 0 0 0 0 0 1 0 0
(e)
Bit number 1 2 3 4 5 6 7 8
Bit change 0 0 1 0 0 0 0 0
(f)
Bit number 1 2 3 4 5 6 7 8
Bit change 0 0 0 0 0 1 0 0
(g)
Bit number 1 2 3 4 5 6 7 8
Bit change 0 0 1 0 0 1 0 0

Figure 31. Option 3 - Key Analysis

When the compressed key (a) is compared with (b), the sixth and eighth bits are different. Option 3 only checks the first bit change and therefore sets a 1 for the sixth bit. The bit-pattern table is initially set to all zeros, and therefore this bit change (d) is used as the bit-pattern table value to produce (f).

Next, the compressed keys (b) and (c) are compared and the bit change (e) is generated in the same manner. The bit-pattern table contains the location of the first change bit for all compressed keys, and therefore (e) and (f) are ORed to produce (g).

The bit changes of all the compressed keys are entered in the bit-pattern table in this manner.

Determining the number of bits to be used for grouping compressed keys

The number of groups or the number of equations is specified by the user through the control statement parameter (CORE, TSIZE). The number of bits in the bit-pattern table that are to be used for grouping the compressed keys is determined as follows:

MIN $(2^N \geq T)$

N: Number of bits to be used for grouping

T: Number of equations determined by the CORE= or TSIZE= specification of the control statement.

As just shown, the number of bits to be used for grouping compressed keys is determined as N that satisfies $2^N \geq T$. That is, the total of N bits that has bit changes in the bit-pattern table (Bit change=1) are used for the grouping.

The actual number of equations for option 3 is determined as 2^N.

An example of this method is shown as follows:

```
Final bit-pattern table
```

Bit number	1	2	3	4	5	6	7	8
Bit change	0	0	1	1	0	1	1	0

```
Number of equations specified by the control statement = 5
```

The bit-pattern table shows that bits 3, 4, 6, and 7 are changed. If this value is substituted in the above equation, N=3 is obtained and the number of equations is $2^3=8$. See Disadvantages for the reason for increasing the number of equations.

Grouping of the compressed keys is performed by checking the changed bits determined above. That is, the first key of a group is selected from where any of the bits 3, 4, or 6 is changed. The following figure shows an example of this.

```
    Number of equations
            specified by the user : 3
            calculated by SRG : MIN(2^N≥3) = 4
    Number of RAPs specified by the user : 8
```

```
 Input key   (1) 0 0 0 0 0 1                          0 0 0 0 0 0 ─┐  Group 1
(Compressed  (2) 0 0 0 0 1 0                          0 0 0 0 0 0 ─┘
 values)     (3) 0 0 0 1 1 0   AND (0 0 1 1 0 0)      0 0 0 1 0 0 ←─  Group 2
             (4) 0 0 1 0 0 0     ──────────→          0 0 1 0 0 0 ─┐
             (5) 0 0 1 0 0 1                          0 0 1 0 0 0  │  Group 3
             (6) 0 0 1 0 1 0   Key comparison,        0 0 1 0 0 0  │
             (7) 0 0 1 0 1 1   up to 2 bits only      0 0 1 0 0 0 ─┘
             (8) 0 0 1 1 0 0                          0 0 1 1 0 0 ←─  Group 4
────────────────────────────────
 Bit changes     0 0 1 1 1 1
(Bit-pattern     ───→          2   = 2
  table)         ────→         2 x 2 = 4       ─────→ Minimum
                 ──────→       2 x 2 x 2 = 8
                 ────────→     2 x 2 x 2 x 2 = 16
```

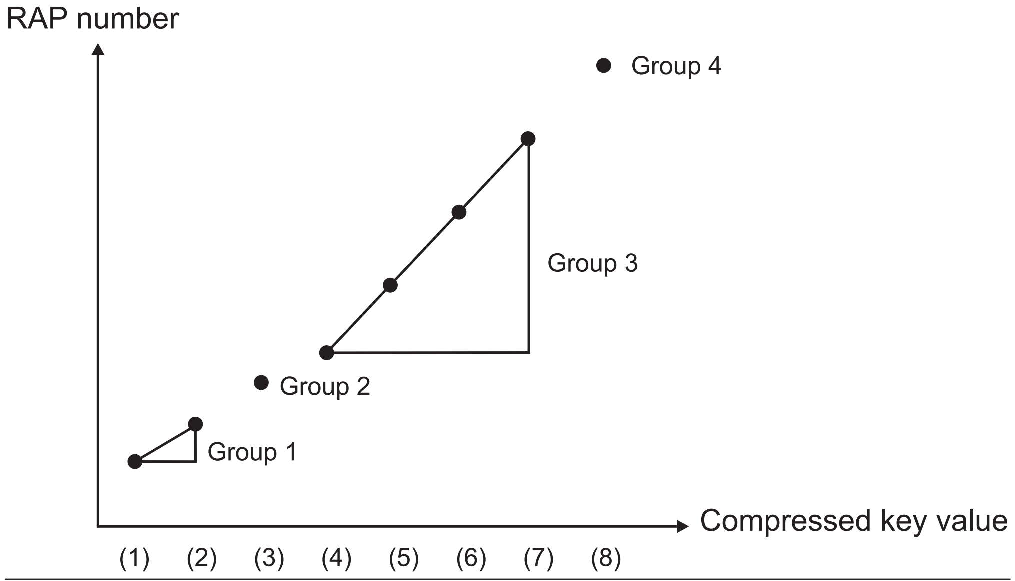


Figure 32. Option 3 - Example 1

The number of entries in the equation table and the total number of RAPs that can be used by the database are calculated from the parameters in the control statement. In this example, the calculated number of equation table entries is 4 (MIN(2^N≥3)) and the total number of RAPs to be used by the database is 8.

With option 3, bit changes of all compressed keys are checked first. In this example, bit changes were at bits 3, 4, 5, and 6. The number of equations obtained above is used to determine which bits are to be used to obtain the equation table entries. The number of bits in this case is equal to the smallest number N that satisfies the condition 2^N≥3, which is 2. Therefore, bits 3 and 4 are used to obtain the equation table entries. The other bits are not used and 0 is set in their bit positions. The results are ANDed to produce 001100. This is performed for all compressed keys, and groups are created where the value changes. In Figure 32, four groups, 1 to 4, were created.

Advantages
The generation of the equation table is faster than with other options because there is no need to read the key file many times or to sort the keys.

Disadvantages
The number of equations may be less than the number specified with the control statement parameter and thus synonyms may occur. The following is an example of this:

Number of equations
specified by the user : 7
calculated by SRG : MIN($2^N \geq 7$) = 8
Number of RAPs specified by the user : 8

```
Input key (compressed values)

      (1) 0 0 0 1 0 0 0 0                       0 0 0 0 0 0 0 0 ┐  Group 1
      (2) 0 0 0 1 0 0 1 0                       0 0 0 0 0 0 0 0 ┘
      (3) 0 0 0 1 0 1 0 0   AND (00101100)      0 0 0 0 0 1 0 0 ←  Group 2
      (4) 0 0 1 0 0 0 0 0     ──────────→       0 0 1 0 0 0 0 0 ┐
      (5) 0 0 1 0 0 0 1 0                       0 0 1 0 0 0 0 0 ┘  Group 3
      (6) 0 0 1 0 0 1 0 0   Key comparison,     0 0 1 0 0 1 0 0 ┐  Group 4
      (7) 0 0 1 0 0 1 0 1   up to 6 bits        0 0 1 0 0 1 0 0 ┘
      (8) 0 0 1 0 1 0 0 0                       0 0 1 0 1 0 0 0 ←  Group 5
────────────────────────────
 Bit        0 0 1 0 1 1 1 1
 changes
(Bit-       ───→              2
 pattern    ───────→          2 x 2 = 4
 table)     ─────────→        2 x 2 x 2 = 8      ──────────→ Minimum
            ───────────→      2 x 2 x 2 x 2 = 16
            ─────────────→    2 x 2 x 2 x 2 x 2 = 32
```

Figure 33. Option 3 - Example 2

In this example, only five groups are created although the number of equations specified is 7. Such a case occurs when the bit changes in the compressed values cluster in the low-order bits.

Key analysis methods summary

The following table summarizes the methods for key analysis.

Table 14. Key Analysis Methods Summary

	OPTION 0	OPTION 1	OPTION 3
Characteristic	The second differences of all the keys are used in analyzing the key distribution. The keys are selected sequentially starting from the one with the largest second difference and are assigned as the first key of each group.	Analysis is performed starting from the leftmost byte of each compressed key.	The bit distribution of all compressed keys is analyzed and the change in bits is used to form groups.
Advantage	If the number of key distribution groups can be estimated, the storage size for the equation table can be specified on the control statement to generate an optimum table without synonyms. If the differences between adjacent keys are all the same, keys can be mapped to RAP numbers with just one equation.	Equation tables can be generated for keys with any distribution.	Generation of equation tables is faster than for other options.
Disadvantage	Synonyms may occur if the keys are scattered.	Compressed keys may be analyzed many times to check each byte and SR (SRR and SRT) generation may take time.	When the bit changes in the compressed key values occur mostly in the low-order bits, the number of equations may be less than the number specified with the control statement parameter and synonyms may occur.

Table 14. Key Analysis Methods Summary (continued)

	OPTION 0	OPTION 1	OPTION 3
Optimum Key File Condition	The number of key groups is predictable and the key intervals are relatively uniform. When grouping this type of keys, there is no need to group by the first byte as with option 1. If the number of key groups is predictable but the key intervals are not uniform, the occurrence of synonyms depends on the TSIZE or CORE parameter specifications. The number of synonyms can be kept small by increasing the value of these parameters. Example: A case where the key is a three-digit branch number plus a four-digit account number with uniform key intervals: 0010001 0010002 : 0010100 0020001 0020002 : 0020100	The number of key groups is predictable and the keys can be grouped by the first byte of each compressed key. If this condition is satisfied and the key intervals within each group are relatively uniform, then the result is similar to option 0. Example: A case where the key is a three-digit branch number plus a four-digit account number with uneven key intervals, but can be grouped by branch number: 1000001 1000003 1000006 1000008 : 1000100 3000003 3000006 : 3000100	The key distribution is random and is difficult to group. Example: 0103254 0105897 0120000 0170582 2013947

Chapter 8. Reference: Migration procedures

The following topics describe migration from an IMS release to another IMS releases, and migration from DBT Version 2 SDO.

Topics:

Migration from an IMS release to another IMS release

If a Sequential Randomizer has been created with SRG in an IMS environment and if you want to use it in a new IMS environment, migrate it to the new IMS environment. This is because changes are made between IMS versions and releases concerning the IMS control blocks that are referred to during the SR execution. SRG provides the functions for the migration.

Procedure

Complete the following steps to migrate:

1. Back up the SR source library, SRRFILE.
2. Define a JCL, using the cataloged procedure FABOSRG. Make sure that the required data sets (for instance, the IMS macro libraries) are the ones required for the new IMS environment.
3. Prepare SRG control statements for the migration. Modify the control statements that were used when the SR was generated, as follows:
   - HDAM SR or PHDAM SR

     Add SRGEN=YES to the ANALYZE control statement. (See "Example 2: Regenerating the SR for an HDAM database" on page 83.)
   - DEDB SR

     If an AREADEF control statement exists, delete it. (See Step 2 of "Example 3: Generating an SR for a DEDB database" on page 84.)

     Make sure that all the DEDB area names are specified in the AREAID parameter of the LINKSR control statement. For the LINKID parameter of the LINKSR control statement, you can select any one of the following options:
     - LINKID=ALL.
     - The LINKID parameter is not specified. (The default option LINKID=ALL is assumed.)
     - The name of the SR is specified on the LINKID parameter. (For the effect of this option, see the description of the LINKID parameter in LINKID.)

       Notes:

       a. The following preconditions must be satisfied:

          The SR source library that was used for generating SR in the old IMS environment exists, and can be used as an input for running SRG in the new IMS environment.
       b. If the LINKID=sr_name option is selected when a DEDB SR is being migrated, the SR load module library that contains the original SR load module must exist and can be used as the input load module library (specified on IMSLIB DD for SRG) for SRG.
4. Run SRG to migrate SR.

Migrating to IMS Version 7 or later from previous releases of IMS

When migrating your existing SR to IMS Version 7 or later, bear the following in mind:

- You need to specify the macro library of High Level Assembler Toolkit on the MACLIB DD of your SRG JCL.
- You can generate a reusable or reentrant HDAM SR.

- If you migrate an HDAM database to a PHDAM database, you must prepare an SR for each partition for new PHDAM database.

Structured programming macro

To assemble SR source files, SRG requires the structured programming macros. In an IMS Version 5 or Version 6 environment, the structured programming macros reside in the MACLIB macro library of IMS. In IMS Version 7 or later, these macros have been removed from the SDFSMAC macro library of IMS and the macro definition file ASMMSP in the High Level Assembler Toolkit is assumed to be used instead. The method of specifying the macro library of High Level Assembler Toolkit is explained in the section headed "JCL procedure" on page 24.

Changing the module attributes of HDAM SR

If you decide to migrate an existing HDAM SR that was generated with nonreusable attribute to an SR that has reusable or reentrant attribute, you must bear in mind that the SR works differently in the IMS online environment. The same consideration applies to SRs for PHDAM database. In some cases, it might be necessary to modify your application program that accesses the HDAM database for which the generated reusable or reentrant SR is used. The same consideration applies to SRs for PHDAM database.

For the details of the differences, see "How the SR works for an HDAM or PHDAM database" on page 76.

Note: Reusable and reentrant attributes are supported for IMS Version 7 or later.

Migrating from HDAM to PHDAM

The general procedure for migrating from an HDAM database to a PHDAM database is described in *IMS Database Administration*. You must use the HALDB Partition Definition Utility to specify the randomizer parameters such as the name of the randomizing module, the number of RAPs per CI or block, or the total number of CIs or blocks in the root addressable area.

Migration from DBT SDO

If you are migrating from DBT Version 2 SDO to SRG under the same IMS release level, it is not necessary to migrate your SRs that were created by DBT Version 2 SDO.

However, if you are upgrading the IMS also, it is necessary to migrate any SRs that were created by DBT Version 2 SDO. The migration procedure is the same as the one described in “Migration from an IMS release to another IMS release” on page 124. Before doing the migration, however, take the following into consideration.

Considerations

The following considerations apply when migrating from DBT SDO.

Cataloged procedure
The name of the cataloged procedure used for SR generation has been changed from DBT Version 2 SDO. Thus, if you are migrating from DBT Version 2 SDO and the old cataloged procedure FABOSDO or PERFSDO is used in your SDO JCL, you must modify the JCL so that the new cataloged procedure FABOSRG will be used.

Problem fix supplied by DBT V2R1 SDO PTF UN14009
You need to take into consideration the maintenance level of the DBT Version 2 with which the SR to be migrated was generated.

The following PTFs were provided for DBT Version 2 SDO:

- PTF UN14009 (APAR PN12944) for DBT Version 2 Release 1 SDO for IMS/ESA® Version 3 Release 1 (FMID: JDBL128)
- PTF UN59997 (APAR PN54805) for DBT Version 2 Release 2 SDO

The fixes provided by these PTFs affect the migration of SDO SRs that have been generated by DBT Version 2 Release 1 SDO without PTF UN14009 applied, or by DBT Version 2 Release 2 SDO without PTF UN59997 applied. When you migrate such an SR to a new IMS environment by doing LINKSR with SRG, you must unload the database by using the original SR and reload it by using the migrated SR. This is necessary, because the migrated SR and the original SR could assign different relative RAP numbers to the same key value.

Migrating your existing DEDB SR of SRTYPE=MULTI

The DEDB SR of SRTYPE=MULTI that is generated by DBT Version 2 SDO uses SDOTABLE to manage DEDB SRs of that type, either CALLTYPE=STD or CALLTYPE=XCI.

In SRG, an SR generated with CALLTYPE=STD still uses the SDOTABLE, but an SR generated with CALLTYPE=XCI no longer uses it. For the details of this change, see “How the SR works for a DEDB” on page 73.

Recommendation: Consider specifying CALLTYPE=XCI to generate a DEDB SR of SRTYPE=MULTI.

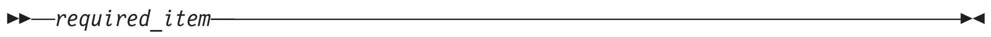
Note that if you want to make an online change to a DEDB SR of SRTYPE=MULTI, you must specify CALLTYPE=XCI.

Messages
SRG does not issue the FABO0115I message for DEDB.

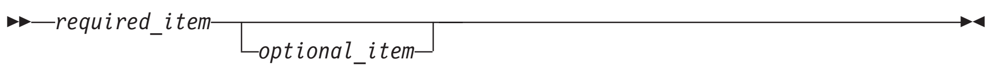
Chapter 9. Reference: How to read syntax diagrams

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

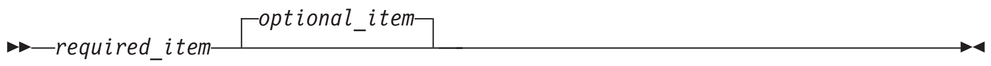
- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following conventions are used:
  - The >>--- symbol indicates the beginning of a syntax diagram.
  - The ---> symbol indicates that the syntax diagram is continued on the next line.
  - The >--- symbol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



- Optional items appear below the main path.

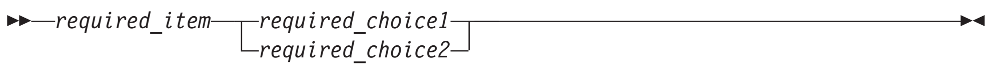


If an optional item appears above the main path, that item has no effect on the execution of the syntax element and is used only for readability.

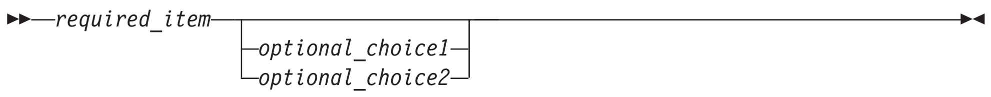


- If you can choose from two or more items, they appear vertically, in a stack.

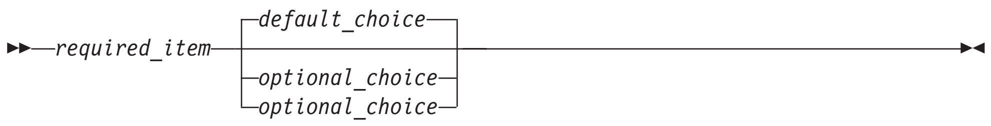
  If you *must* choose one of the items, one item of the stack appears on the main path.



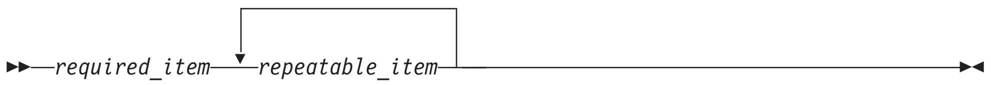
If choosing one of the items is optional, the entire stack appears below the main path.



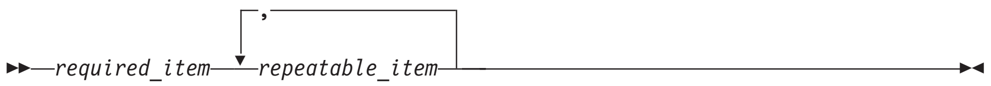
If one of the items is the default, it appears above the main path, and the remaining choices are shown below.



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



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



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

- Keywords, and their minimum abbreviations if applicable, appear in uppercase. They must be spelled exactly as shown. Variables appear in all lowercase italic letters (for example, *column-name*). They represent user-supplied names or values.
- Separate keywords and parameters by at least one space if no intervening punctuation is shown in the diagram.
- Enter punctuation marks, parentheses, arithmetic operators, and other symbols exactly as shown in the diagram.
- Footnotes are shown by a number in parentheses; for example, (1).

Chapter 10. Troubleshooting

Use the following information to troubleshoot IMS Sequential Randomizer Generator problems.

Topics:

Messages and codes

The following topics describe the messages and return codes issued by SRG (during the SR generation) and by the SR (during IMS execution).

The following messages, return codes, and abend codes are described:

- SRG abend codes
- SRG return codes
- SR messages (issued by the generated SR during IMS execution)
- SRG messages (issued by SRG during the SR generation)
- SRG control statement messages (issued by SRG when the control statements are analyzed).

Note: For the SR return codes, see "How the Sequential Randomizer (SR) works" on page 73.

The SR messages and SRG messages have the following format:

```
FABOnnnnx  text
```

Where:

FABO Indicates that the message was issued by IMS Sequential Randomizer Generator

nnnn Indicates the message identification number

x Indicates the severity of the message:

- **A** Indicates that operator intervention is required before processing can continue.
- **E** Indicates that an error occurred, which might or might not require operator intervention.
- **I** Indicates that the message is informational only.
- **W** Indicates that the message is a warning to alert you to a possible error condition.

Each message also includes the following information:

Explanation:
The Explanation section explains what the message text means, why it occurred, and what its variables represent.

System action:
The System action section explains what the system will do in response to the event that triggered this message.

User response:
The User response section describes whether a response is necessary, what the appropriate response is, and how the response will affect the system or program.

SRG abend codes

SRG uses only one abend code; 3999. This abend code is accompanied by the FABO3999A message. If SRG ends with this abend code, contact IBM Software Support.

SRG return codes

SRG returns the following three types of return codes according to the results of the SR generation:

Code	Meaning
0	Successful completion. No errors were detected. An SR was generated.
4	The meaning of this return code depends on the database type: • In DEDB, if the LINKSR control statement was specified, return code 4 means that an error occurred while processing the specified area, but that an SR was generated. If the LINKSR control statement was not specified, the return code indicates that an error occurred during the processing of the specified area and SR was not generated. • In HDAM and PHDAM, this return code means that an error occurred during SRG processing, but that an SR was generated.
8	An error occurred during SRG processing and the process could not be continued. SR was not generated.

SR return codes and reason codes

SR returns the following return codes and reason codes:

HDAM SR and PHDAM SR

Code	Meaning
0	Normal end. No errors were detected.
4	See Table 6 on page 78.
8	See Table 6 on page 78.

DEDB SR (CALLTYPE=STD)

Code	Meaning
0	Normal end. No errors were detected.
4	See Table 7 on page 79.
8	See Table 7 on page 79.

DEDB SR (CALLTYPE=XCI)

DEDB SR of CALLTYPE=XCI issues the following return codes. The return code 2*xx* might be issued at the initialization call, and the return code 3*xx* might be issued at the termination call to SR.

Code	Meaning
0	Normal end. No errors were detected.
4	At the randomizing call, this return code notifies IMS that some error occurred while the call was being processed and that the status code "FM" should be returned to the application program. See also Table 7 on page 79. At the initialization call or termination call, this return code means that some interface error was detected. SR returns the following reason codes and IMS issues the DFS2627I message (for the initialization call) or DFS2628I message (for the termination call):

Reason Code (HEX)
Meaning

00000001
The non-XCI randomizer is called with the extended call interface.

00000002
An unexpected value was passed in Register 0.

8 At the randomizing call, this return code notifies IMS that some error occurred while the call was being processed and the SR cannot continue processing. If IMS receives this return code, IMS forces the dependent region that issued the call to abend. See also Table 7 on page 79.

At the initialization call or termination call, this return code means that an unexpected entry code was passed. IMS issues the DFS2627I message (for the initialization call) or DFS2628I message (for the termination call). The reason code that is shown in the message is the entry code that was passed to the SR.

12 Invalid user data is passed. The reason code that is shown in the message is the user data that has been passed.

16 Incorrect environment ID is passed. The reason code that is shown in the message is the environment ID that has been passed.

20 This code might be returned at the initialization call or the termination call. It indicates that an error occurred while the IMS Callable Service was being initialized. IMS issues the DFS2627I message (for the initialization call) or the DFS2628I message (for the termination call). The reason code that is shown in the message is the error reason code that was returned from DFSCSII0.

30 This code might be returned at the initialization call or the termination call. It indicates that the SR failed to get the work area. If SR is running under the IMS environment, IMS issues the DFS2627I message (for the initialization call) or the DFS2628I message (for the termination call). The reason code that is shown in the message is the error reason code that was returned from DFSCSIF0. If SR is running under the MVS batch environment, SR issues message FABO0010A. The reason code that is shown in the message is the system completion code that was returned from the LOAD macro.

206 SR failed to load the extended master table (SRT). If SR is running under the IMS environment, IMS issues message DFS2627I. The reason code that is shown in the message is the error reason code that was returned from DFSCSIF0. If SR is running under the MVS batch environment, SR issues message FABO0011A. The reason code that is shown in the message is the system completion code that was returned from the LOAD macro.

207 SR failed to load an equation table. If SR is running under the IMS environment, IMS issues message DFS2627I. The reason code that is shown in the message is the error reason code that was returned from DFSCSIF0. If SR is running under the MVS batch environment, SR issues message FABO0011A. The reason code that is shown in the message is the system completion code that was returned from the LOAD macro.

208 SR failed to load an equation table and failed to delete the extended master table. If SR is running under the IMS environment, IMS issues message DFS2627I. The reason code that is shown in the message is the error reason code that was returned from DFSCSIF0. If SR is running under the MVS batch environment, SR issues message FABO0011A. The reason code that is shown in the message is the return code that was returned from the DELETE macro.

209 SR failed to load an equation table and also failed to delete the equation tables that have been already loaded. In this case, the extended master table is not deleted. If SR is running under the IMS environment, IMS issues message DFS2627I. The reason code that is shown in the message is the error reason code that was returned from DFSCSIF0. If SR is running under the MVS batch environment, SR issues message FABO0011A. The reason code that is shown in the message is the return code that was returned from the DELETE macro.

306 SR failed to delete an equation table. If SR is running under the IMS environment, IMS issues message DFS2628I. The reason code shown in the message is the error reason code that was returned from DFSCSIF0. If SR is running under the MVS batch environment, SR issues message FABO0011A. The reason code that is shown in the message is the return code that was returned from the DELETE macro.

307 SR failed to delete the extended master table. If SR is running under the IMS environment, IMS issues message DFS2628I. The reason code that is shown in the message is the error reason code that was returned from DFSCSIF0. If SR is running under the MVS batch environment, SR issues message FABO0011A. The reason code that is shown in the message is the return code that was returned from the DELETE macro.

SR messages

The Sequential Randomizer (SR) created by SRG displays messages on the OS console during IMS run.

FABO0001A INPUT KEY LENGTH IS UNMATCH WITH SRR

Explanation: This message is issued to the OS console when, during IMS execution, the input key length was different from the one specified at the SR generation.

System action:

- DEDB: The dependent region that issued the DL/I call ends abnormally with abend U1021.
- HDAM or PHDAM (VSAM or OSAM): The application program that issued the DL/I call ends abnormally with abend U812.

Operator response: Report the error to the IMS database administrator.

User response: Check to see if a wrong SR name was specified on the DBDGEN control statement or a wrong key length was specified at DBDGEN. Correct the errors and rerun the program.

FABO0002A INVALID KEY PASSED TO SR=*xxxxxxxx*. KEY TYPE ERROR

Explanation: This message is issued to the OS console when the attribute of an input key was different from the one used at the SR generation. *xxxxxxxx* is the name of SR that issued the message.

System action:

- DEDB: When ERPROC=CONT is specified on the ANALYZE control statement, SR returns RC=4 to IMS and IMS returns the status code 'FM' to the application program that issued the DL/I call, and the corresponding dependent region continues processing. When ERPROC=CONT is not specified, the dependent region that issued the DL/I call ends abnormally with abend U1021.
- HDAM or PHDAM (VSAM or OSAM):
  - When ERPROC=CONT is specified on the ANALYZE control statement, SR returns RC=X'04' to IMS, and IMS returns the status code "FM" to the application program that issued the DL/I call, and the corresponding dependent region continues processing.
  - When ERPROC=ABEND is specified or the ERPROC parameter is not specified, the application program that issued the DL/I call ends abnormally with abend U812.

Operator response: Report the error to the IMS database administrator.

User response: Check to see if a wrong SR name was specified on the DBDGEN control statement or if a DL/I call was issued by the application program without checking the key attribute. Correct the SR name of DBDGEN, or add the attribute check function to the application program, and rerun the program.

FABO0003A INVALID KEY PASSED TO SR=*xxxxxxxx*. OUT OF AREA RANGE

Explanation: This message is issued to the OS console when a key that did not exist within the key range for any of the SRT areas was given to the SR for DEDB. *xxxxxxxx* is the SR name that issued the message.

System action: When ERPROC=CONT is specified in the ANALYZE control statement, SR returns RC=4 to IMS and IMS returns the status code 'FM' to the application program that issued the DL/I call, and the corresponding dependent region continues processing. When ERPROC=CONT is not specified, the dependent region that issued the DL/I call ends abnormally with abend U1021.

Operator response: Report the error to the IMS database administrator.

User response: Correct the key if it is incorrect, or check if the area for the key exists in SRT. If not, code an AREADEF control statement to create the equation table for the area, add a new area name to the AREAID parameter of the LINKSR control statement in order to merge the above equation table with the existing tables, and run the SRG job. Then, rerun the program.

FABO0005A GETMAIN FAILURE AT *xxxxxxxx*

Explanation: SR for DEDB failed to acquire a work area in ECSA. *xxxxxxxx* is the SR name that issued the message.

System action: The dependent region that issued the DL/I call ends abnormally with abend U1021.

Operator response: Report the error to the IMS database administrator.

User response: Prepare a work area of 2048 bytes in ECSA and rerun the program.

FABO0006A SRT LOAD FAILURE RC=*aa* REASON=*bb* AT xxxxxxxx

Explanation: This message is issued to the OS console when a DEDB SR of multiple module type failed to load a module or failed to acquire or delete a virtual storage using the IMODULE service of IMS. *xxxxxxxx* indicates the SR name which issued this message. *aa* indicates the return code from the IMODULE service at the time of error.*bb* indicates the function of IMODULE service which was used at that time. See the following table:

IMODULE service	Reason codes (bb)
GETMAIN	G1, G2, G3
LOAD	L1, L2, L3
DELETE	D1, D2, D3, D4, D5

System action: The dependent region that issued the DL/I call ends abnormally with abend U1021.

Operator response: Report the error to the IMS database administrator.

User response: The following reasons should be considered depending on the value of *bb*:

bb=**'Lx':**

- The extended master table or an area equation table could not be found in the data set described in the STEPLIB of the JCL which started the IMS control region.
- The storage for the ECSA is not large enough to load the extended master table or an area equation table.

bb=**'Gx':**
The storage for the ECSA is not large enough to acquire the area for a table used by the SR.

bb=**'Dx'** Refer to the return code *aa* of the IMODULE service.

FABO0007A LOAD FAILURE FOR *xxxxxxxx* **AT** *yyyyyyyy*

Explanation: This message is issued to the OS console when a DEDB SR of multiple module type failed to load the extended master table or an area equation table while it is used by the DBT Version 2 DEDB Unload/Reload utility or by DEDB Tuning Aid utility. *xxxxxxxx* is the name of the table which the SR could not load. *yyyyyyyy* is the name of the SR which issued the message.

System action: The SR returns RC=8 to the caller.

Operator response: Report the error to the IMS database administrator.

User response: The following reasons should be considered:

- The extended master table or an area equation table indicated by *xxxxxxxx* could not be found in the STEPLIB for the job.
- The storage for the extended private area is not large enough to load the table *xxxxxxxx*.

FABO0008A NON-XCI CALL TO XCI RANDOMIZER *xxxxxxxx*

Explanation: This message is issued to the OS console when an XCI SR is called by IMS with the standard (non-XCI) interface.

System action: The SR returns a return code of 8 to the caller. If the SR is used in the IMS online environment, the dependent region that issued the DL/I call ends abnormally with the abend U1021. If the SR is used in an MVS batch job, the result depends on the program that called the SR.

Operator response: Report the error to the IMS database administrator.

User response: If the SR was used in the IMS online environment, check if the SR was defined as an XCI randomizer in DBDGEN. If not, define it. If you cannot determine the cause of the problem, save any dump and SYSLOG information, and contact the IBM Support Center.

If the SR is used in an MVS batch job, check your program logic. If you cannot determine the cause, save any dump and contact the IBM Support Center.

FABO0009A RANDOMIZER *sr_name* **IS NOT INITIALIZED**

Explanation: The XCI SR *sr_name* has been called for randomizing, but the user data passed from IMS is null which means that the SR was not initialized.

System action: The SR returns a return code of 8 to IMS. The dependent region that issued the DL/I call ends abnormally with the abend U1021.

Operator response: Report the error to the IMS database administrator.

User response: Check why the SR was not initialized before the first randomizing call was issued. If you cannot determine the cause of the problem, save any dump and SYSLOG information, and contact the IBM Support Center.

FABO0010A ERROR IN [INITIALIZING | TERMINATING] XCI RANDOMIZER -*rrrrrrrr* **- FAILED IN GETTING WORKAREA**

Explanation: The SR for DEDB failed to acquire a work area during batch XCI initialization or termination. *rrrrrrrr* is the SR name that issued the message.

System action: The SR returns a return code of 30 and a reason code equal to the return code of the GETMAIN macro to the caller.

Operator response: Report the error to the IMS database administrator.

User response: Increase the region size of the batch job.

FABO0011A ERROR IN INITIALIZING|TERMINATING XCI RANDOMIZER -*rrrrrrrr* - FAILED IN LOADING|DELETING *tttttttt* RETURN CODE=*xxxx* REASON CODE=*yyyyyyyy*

Explanation: The SR for DEDB failed to load or delete the extended master table or an area equation table during batch XCI initialization or termination. *rrrrrrrr* is the SR name that issued the message. *tttttttt* is the name of the extended master table or an area equation table that could not be loaded or deleted. *xxxx* is the return code and *yyyyyyyy* is the reason code of the LOAD or the DELETE macro that caused the error.

System action: The SR returns a return code and a reason code, which are described in "SR return codes and reason codes" on page 131.

Operator response: Report the error to the IMS database administrator.

User response: Check the return code and the reason code to determine the cause of the problem. If you cannot determine the cause of the problem, contact IBM Software Support.

FABO0051A ATTRIB=REUS/RENT IS NOT SUPPORTED IN THIS IMS RELEASE

Explanation: The SR is link-edited with reusable or reentrant attribute with ATTRIB=REUS or =RENT specified on the ANALYZE control statement when it was generated; also, it is used in an IMS environment earlier than IMS Version 7. The reusable and reentrant SR is supported only in Version 7 or later.

System action: The SR returns a return code of 8 to the caller. If the SR is used in the IMS online environment, the dependent region that issued the DL/I call ends abnormally with abend U812.

Operator response: Report the error to the IMS database administrator.

User response: Check the IMS environment where the SR was generated. If a wrong SR is used, use the correct one. If you intend to use the SR in an IMS environment earlier than IMS Version 7, you must regenerate the SR in that environment without specifying the ATTRIB parameter in the ANALYZE control statement.

FABO0052A GETMAIN FAILURE AT *sr_name*

Explanation: The reusable or reentrant SR *sr_name* for HDAM or PHDAM failed to allocate a work area in ECSA. An SRG SR that has the reusable or reentrant attribute allocates a 2,048-byte work area for each dependent region in which a DL/I call is issued for the database or partition for which the SR is used.

System action: The SR returns a return code of 8 to the caller. If the SR is used in the IMS online environment, the dependent region that issued the DL/I call ends abnormally with the abend U812.

Operator response: Report the error to the IMS database administrator.

User response: Prepare a storage area of 2,048 bytes in ECSA.

SRG messages

The following topics provide IMS Sequential Randomizer Generator messages.

FABO0101I ALL SRG PROCESS COMPLETED

Explanation: SRG processing has been completed successfully with RC=0, and the SR has been generated.

System action: SRG generates the SR load module and stores it in the data set specified with the IMSLIB parameter of the cataloged procedure FABOSRG.

User response: None. This message is informational.

FABO0102I SUCCESSFUL ENDING OF FABOMAIN BUT SR NOT GENERATED

Explanation: SRG processing has been completed normally, but no SR load module has been generated. This message is issued only when TYPE=DEDB was specified on the ANALYZE control statement and the LINKSR control statement was not coded. This message is not issued when TYPE=OSAM or VSAM.

System action: The SRG job ends with RC=4 because no SR load module is generated.

User response: This message is issued when the SR load module is not generated intentionally (TYPE=DEDB and no LINKSR control statement coded). When the SR load module is to be generated, examine the FABO0105I message issued before this message. If FABO01051 message is issued for all the area names specified as the label of the AREADEF control statements, the DEDB area processing is not necessary when generating the SR load module. (See “Control statements” on page 28.) Of the areas to which the FABO0105I message is not issued, analyze the messages issued just after the FABO0104I message and rerun the area processing.

FABO0103I UNSUCCESSFUL END OF SRG PROCESS

Explanation: An abnormal status occurred during SRG processing. This message is preceded by the messages describing each status. Causes that may have prevented the SRG execution are as follows:

- There was a specification error on a control statement.
- The SRG load module did not exist.
- All the DEDB area processes were ends abnormally. (DEDB)
- The SR source library did not contain all the DEDB area source specified on the AREAID parameter of LINKSR control statement. (DEDB)
- I/O error occurred on a data set. (HDAM)

System action: SRG ends with RC=8 (unable to continue processing) after issuing messages which describe the errors.

User response: Analyze the error from the messages issued before this message, take corrective actions, and rerun the SRG job.

FABO0104I PROCESS FOR AREA = ***xxxxxxxx*** **STARTED**

Explanation: Processing of the area name *xxxxxxxx* started. *xxxxxxxx* is the name specified in the AREADEF control statement label. This message is issued only when TYPE=DEDB was specified on the ANALYZE control statement and also an AREADEF control statement was coded.

System action: Processing continues.

User response: If this message is followed by the FABO0105I message, it means that this area processing has been completed normally. If it is followed by the other messages, it indicates that the area processing ended abnormally by an error. See the error messages, if any, to find the cause of the error. Correct the error accordingly, and rerun the SRG job only for the area reported in the message.

FABO0105I PROCESS FOR AREA = ***xxxxxxx*** **COMPLETED SUCCESSFULLY**

Explanation: Processing of the area name *xxxxxxxx* has been completed normally in SRG. *xxxxxxxx* is the name specified as the label field of the AREADEF control statement. This message is issued only when TYPE=DEDB was specified on the ANALYZE control statement and also an AREADEF control statement was coded.

System action: Processing continues.

User response: If this message follows the FABO0104I message, it indicates that the area processing has been completed normally. If this message does not follow FABO0104I message, it means the area processing ended abnormally by an error. If a message other than FABO0105I is issued after FABO0104I, that message shows the cause of the error. The user must find the cause of the error, take corrective actions, and then rerun SRG only for the area reported in the message.

FABO0106A AREADEF AND/OR LINKSR NOT SPECIFIED

Explanation: Although TYPE=DEDB was specified on the ANALYZE control statement, neither the AREADEF nor the LINKSR control statement was coded. If TYPE=DEDB, AREADEF and/or LINKSR control statements must be coded after the ANALYZE control statement.

System action: SRG processing ends with RC=8.

User response: Corrective actions depend on the database type as follows:

- DEDB: Code AREADEF and/or LINKSR control statements after the ANALYZE control statement and rerun the SRG job.
- HDAM and PHDAM (VSAM or OSAM): Change the TYPE parameter of the ANALYZE control statement to VSAM or OSAM and rerun the SRG job.

FABO0107A CONTROL STATEMENT ERROR

Explanation: One of the following occurred:

- A control statement other than ANALYZE, AREADEF, or LINKSR was specified.
- An incorrect or conflicting specification was found in a parameter of the ANALYZE, AREADEF, or LINKSR control statement.

The details of the error are obtained in the control statement assembly listing by the MNOTE macro message. (See SRG control statement messages.)

System action: SRG ends processing with RC=8.

User response: Correct the error that is indicated in the assembly listing and rerun the SRG job. (The error messages for the control statements are explained in “SRG control statement messages” on page 152.) The message, "END STATEMENT MISSING," for the control statement may be ignored.

FABO0108I GENERATED SR SIZE IS *xxxxxxxx* BYTES

Explanation: This message shows the size of the generated SR load module in bytes. *xxxxxxxx* indicates the total size of SRR and SRT.

System action: SRG continues processing.

User response: If the *xxxxxxxx* value is extremely large compared with the value specified with the TSIZE or CORE parameter of the control statement, it means that the keys of the input key file were not grouped correctly. Increase the value of the TSIZE or CORE parameter and rerun the SRG job.

FABO0109A RANGE OF AREA IS OVERLAPPED

Explanation: The key ranges that are specified with the KEYRNG or ARETRY parameter of the AREADEF control statement for two or more areas overlap.

In the following example, the keys 050 - 070 overlap between areas AR1 and AR2:

```
AR1   AREADEF   KEYRNG = (001, 070)
AR2   AREADEF   KEYRNG = (050, 100)
```

System action: SRG ends processing with RC=8.

User response: Check the key ranges for the areas to be used by the database. If an error is found in the specification of the key range (specified with the KEYRNG or ARETRY parameter of the AREADEF control statement), the equation table must be re-created for the area in which the error was found. In the previous example, if the key range for area AR1 is incorrect and the key range for AR2 is correct, the SRG job must be rerun after the value of the KEYRNG or ARETRY parameter of the AREADEF control statement for AR1 has been corrected. The AREADEF control statement for AR2 does not need to be specified.

FABO0110A FILE OPEN OR I/O ERROR DDN = *xxxxxxxx*

Explanation: One of the following occurred:

- The data set to be used by SRG did not exist, or DUMMY was specified on the DD statement.
- An I/O error occurred when a GET/PUT was run for a data set to be used by SRG.

xxxxxxxx is the ddname of the data set in which the error occurred.

System action: Processing depends on the database type that was specified on the ANALYZE control statement.

- If the database type is HDAM or PHDAM (VSAM or OSAM), SRG ends processing with RC=8.
- If the database type is DEDB, SRG starts processing of the next area.

User response: Take one of the following actions depending on the error type:

- When the DD statement indicated by *xxxxxxx* does not exist or a DUMMY is specified on it, define the DD statement by referring to the cataloged procedure FABOSRG and then rerun the SRG job.
- When an I/O error occurred on a data set, make a copy of the data set on a different volume and then rerun the SRG job.

FABO0111A MODULE LINK ERROR MEMBER = *xxxxxxxx* NOT FOUND IN DDN = STEPLIB

Explanation: One of the following occurred:

- The module required for SRG execution was not found in the data set specified by DDN=STEPLIB. *xxxxxxxx* is the load module name that was not found (MEMBER=FABOxxxxx is shown).
- When a compression routine or an SR load module was generated by SRG, the load module of the compression table (table for checking the attribute of the input key and for calculating the compressed value) was not found in the data set specified by DDN=STEPLIB. *xxxxxxxx* is 'IEWL'.

- The REGION size was insufficient for SRG execution. SRG could not load the module *xxxxxxxx*.

System action: SRG is unable to continue processing and ends with RC=8.

User response: Check to see if the necessary library is included as STEPLIB or sufficient REGION size is provided for SRG. In the former case, add the necessary library to STEPLIB and rerun the SRG job. In the latter case, increase the REGION size specified in the cataloged procedure FABOSRG and rerun the SRG job.

FABO0112I COMPRESSION ROUTINE OF *xxxxxxxx* GENERATED

Explanation: The load module of the compression routine was generated correctly by SRG execution. This message is issued to the OS console by WTO. *xxxxxxxx* is the SR name specified in the label field of the ANALYZE control statement.

System action: SRG continues processing.

Operator response: None.

User response: None. This message is informational.

FABO0113I OPTION PROCESS OF *xxxxxxxx* COMPLETED

Explanation: This message indicates that SRG completed the database key analysis correctly. This message is issued to the OS console by WTO. When the database type is HDAM or PHDAM (VSAM or OSAM), *xxxxxxxx* is the SR name specified in the ANALYZE control statement label. When the database type is DEDB, *xxxxxxxx* is the area name specified in the AREADEF control statement label.

System action: SRG continues processing.

Operator response: None.

User response: None. This message is informational.

FABO0114I OPTIMIZATION PROCESS OF *xxxxxxxx* COMPLETED

Explanation: This message indicates that the optimization conditions (maximum number of synonyms and maximum CI capacity ratio) specified on the control statement were satisfied. This message is issued to the OS console by WTO. When the database type is HDAM or PHDAM (VSAM or OSAM), *xxxxxxxx* is the SR name specified in the ANALYZE control statement label. When the database type is DEDB, *xxxxxxxx* is the area name specified in the AREADEF control statement label.

System action: SRG continues processing.

Operator response: None.

User response: None. This message is informational.

FABO0115I RAP NUMBER CHANGED. REFER TO RECOMMENDED DBD

Explanation: This message is issued when SRG has automatically increased the number of RAPs of HDAM or PHDAM. SRG increases the number of RAPs when, as a result of an input key analysis or the optimization process, more RAPs are needed than are specified in the control statement.

The initial total number of RAPs is calculated from certain parameters of the ANALYZE control statement as follows:

```
number of RAPs = a x b x c x d
```

where:

-

a	is the number of RAPs per CI, which equals the quotient of: `CIBYTE x (max CI capacity ratio) / (average record length)` See Chapter 5, "Reference: How the Sequential Randomizer is generated," on page 95 for the details.
b	is the number of blocks or CIs per track (CITRK)
c	is the number of tracks per cylinder (TRCYL)
d	is the number of cylinders (PRIMCI)

System action: SRG continues processing.

User response: First, you must check the recommended DBDGEN parameters in the "Recommended DBD Control Statement report" on page 68.

If the total number of RAPs has been changed, but you do not want to change the original value, take the following actions and run the SRG job again:

- If the OPTMZ parameter is not specified, increase the number of blocks or CIs in the root addressable area, changing the value for the PRIMCI parameter, so that the number of RAPs can be enough.
- If the OPTMZ parameter is specified:
  - Relax the optimization conditions, especially the maximum CI capacity ratio.
  - Increase the size of the equation table, using the CORE parameter.

If the increased number of RAPs is within the allowable range, use the DBDGEN parameters shown on the SR assembly listing and run your DBDGEN job.

FABO0116A LOGIC ERROR FOUND IN SRG

Explanation: This message is issued in the following cases:

- An error was found during the assembling or link-editing of the compression routine or SR, and processing was ended. (RC=4-16 in assemble and RC=4-12 in link-edit)
- A logic error was found during SRG processing.

System action: SRG ends processing with RC=8.

User response: Specify the SYSOUT class on the SYSDUMY DD statement, rerun the SRG job, and check the error message of the assembler or the link-edit program. Correct the error and rerun the SRG job. If no SRG error message is issued after the rerun, contact IBM.

FABO0117I WARNING MESSAGE ISSUED, BUT SR IS GENERATED

Explanation: SRG issued a warning message, but SR was generated.

System action: SRG continues processing, and eventually ends with RC=4.

User response: See the Programmer Response of the message that was issued before this message.

FABO0118A *xxxxxxxx* TERMINATED ABNORMALLY

Explanation: An error was found during the assembling or link-editing of the compression routine or SR, and processing was ended. (RC=20 or greater in assembling, and RC=16 or greater in link-editing.) *xxxxxxxx* is either ASSEMBLE or LINKEDIT.

System action: SRG ends processing with RC=8.

User response: Specify the SYSOUT class on the SYSDUMY DD statement, rerun the SRG job, and check the error message of the assembler or link-edit program. Correct the error and rerun the SRG job.

FABO0119A THE STORAGE FOR COMPRESSED KEY FILE COULD NOT BE GETMAINED, KEYNBR : *xxxxxxxx*

Explanation: GETMAIN failed for the compressed key file in the extended private area. *xxxxxxxx* is the value specified in the KEYNBR parameter of the ANALYZE control statement.

Storage required for the generation of the compressed key file can be calculated as follows:

```
(INKEYL + CKEYL + 2) x KEYNBR
```

where:

INKEYL
The length of input key, which is specified in the INKEYL parameter of the ANALYZE control statement

CKEYL The length of the key compressed by SRG.

System action: SRG ends processing with RC=8.

User response: Check the value specified in the KEYNBR parameter, acquire the storage, and rerun the SRG job. If the required storage cannot be acquired by GETMAIN, move out the KEYNBR parameter specification and rerun the job again. In this case, the compressed key file will be generated on DASD.

FABO0120I SRT SIZE EXCEEDED TLIMIT(,AREA:*xxxxxxxx*)

Explanation: SRT size reached the maximum value specified by the TLIMIT parameter during the equation table optimization. *xxxxxxxx* is the name of the DEDB area (in the case of DEDB) for which this message is issued. In the case that TLIMIT parameter was not specified, this message is issued when the number of equations in an SRT reached 32767.

System action: SRG ends the optimization process, and continues processing.

User response: To further optimize the equation table, even if the number of equations in an SRT exceeds 32767, use the TNUMLIM parameter instead of the TLIMIT parameter.

FABO0121A SR LOAD MODULE IS NOT FOUND IN IMSLIB THOUGH LINKID=ALL IS NOT SPECIFIED

Explanation: Some (but not all) areas were selected in the LINKID parameter for the LINKSR processing, but the SR load module did not exist in IMSLIB.

System action: LINKSR processing is canceled. SRG does not generate the SR load module, and ends the job with RC=8.

User response: If the SR load module library (IMSLIB) is not specified correctly, run SRG again with the correct IMSLIB specification.

If all the DEDB areas on the AREAID parameter should be processed, run SRG again without LINKID parameter or with LINKID=ALL.

FABO0122I SR WAS GENERATED AS MULTIPLE MODULES: SR=*SR_name*

Explanation: A sequential randomizer has been generated successfully as multiple modules with SRR name=*SR_name*.

System action: SRG generates the SR as multiple modules.

User response: None. This message is informational.

FABO0123I TYPE CHANGE TO ***SR_type*** **COMPLETED FOR SR=*****SR_name***

Explanation: TYPECHG=YES is processed normally and the type of sequential randomizer *SR_name* has been changed to *SR_type* successfully, where *SR_type*=SINGLE or MULTI.

System action: SRG updates the SRR source of the SR, and changes the module type of the SR.

User response: None. This message is informational.

FABO0124E INVALID PARAMETER IN PARM FIELD OF EXEC STATEMENT

Explanation: An incorrect value was specified for a parameter in the PARM field of EXEC JCL statement, or the format of the PARM field was incorrect.

System action: SRG issues the FABO0103I message and ends with RC=8.

User response: Specify the correct values on the PARM field of EXEC statement and rerun SRG.

FABO0125E INVALID VALUE SPECIFIED FOR KEYWORD: *xxxxxxxx*

Explanation: An incorrect value was specified for the parameter *xxxxxxxx* in the PARM field of EXEC JCL statement.

System action: SRG issues the messages FABO0124E and FABO0103I and ends with RC=8.

User response: Specify the correct value for the parameter *xxxxxxxx* and rerun SRG.

FABO0126I SR LINK-EDIT RC=00, BUT CHECK THE OUTPUT FROM THE LINKAGE EDITOR

Explanation: If you specified LNKPGM=LKED (default) or LNKPGM=BINDERLKED in the ANALYZE control statement, and if the DFSMS/MVS Binder returned the return code of 0 at the LINKSR process (for DEDB) or the SRGEN process (for HDAM and PHDAM), this information message is issued. If COMPAT(LKED) option is specified, the DFSMS/MVS Binder returns the return code of 0 if the reusability attribute of a module was downgraded than that was specified in the Binder option. If you have link-edited an SR outside the SRG environment, and if the reentrant or reusable attribute has been accidentally removed from the SR, the above problem can occur.

System action: SRG continues processing and ends with RC=0.

User response: See the output from the Binder and check the addressing mode, the residency mode, and the reusability attribute of the generated SR. For the attributes of a DEDB SR, see “Generating an SR load module” on page 105 in Chapter 5, “Reference: How the Sequential Randomizer is generated,” on page 95. If your DEDB SR was accidentally link-edited with incorrect attributes, you can recover the original attributes by running a LINKSR job with LNKPGM=BINDER specified.

FABO0127W SR LINK-EDIT RC=04, CHECK THE OUTPUT FROM LKED/BINDER (LINKSR ***n*****)**

Explanation: The DFSMS/MVS Binder returned the return code of 4 at the LINKSR process or at the SRGEN process. The number *n* indicates one of the following LINKSR-process types:

LINKSR 0
The LINKSR process for a DEDB SR of the single-module type, or the SRGEN process for an HDAM or PHDAM SR.

LINKSR 1
The LINKSR process for the AREA equation tables and the logic part of a DEDB SR of the multiple-module type.

LINKSR 2
The LINKSR process for the extended master table of a DEDB SR of the multiple-module type.

System action: SRG continues processing.

User response: Check the output from the Binder for the indicated LINKSR or SRGEN process. If you have done the LINKSR for a DEDB SR that was generated by the SDO of the level before APAR PN82345 for DBT Version 2 Release 2 SDO, the following messages issued from the Binder can be ignored:

- IEW2646W for the SRR CSECT or for the AREA equation table CSECTs
- IEW2651W for the SRR CSECT or for the AREA equation table CSECTs
- IEW2400I for the TBLGET CSECT, the BRGTAB*x* (*x*=1 to 8) CSECTs, and the FABOTAB*x* (*x*=1 to 8) CSECTs

FABO0127A SR LINK-EDIT RC=***xx*****, CHECK THE OUTPUT FROM LKED/BINDER (LINKSR** ***n*****)**

Explanation: The DFSMS/MVS Binder returned the return code of *xx* at the LINKSR or SRGEN process. The number *n* indicates one of the following LINKSR-process types:

LINKSR 0
The LINKSR process for a DEDB SR of the single-module type, or the SRGEN process for an HDAM or PHDAM SR

LINKSR 1
The LINKSR process for the AREA equation tables and the logic part of a DEDB SR of the multiple-module type

LINKSR 2
The LINKSR process for the extended master table of a DEDB SR of the multiple-module type

System action: SRG ends with RC=8.

User response: Check the output from the Binder and correct the errors.

FABO0128I GENERATED SR *sr_name* **HAS** *xxxxxxxxxxx* **ATTRIBUTE**

Explanation: An HDAM or PHDAM SR *sr_name* is created, and the attribute of the generated SR module is *xxxxxxxxxxx*.

System action: SRG continues processing.

User response: None. This message is informational.

FABO0129I GENERATED SR *sr_name* **IS AN XCI RANDOMIZER**

Explanation: A DEDB SR *sr_name* is created, and it is an XCI randomizer.

System action: SRG continues processing.

User response: You must define the SR as an XCI randomizer in the DBDGEN.

FABO0130W DBLEN PARAMETER IS OBSOLETE - ACTION REQUIRED

Explanation: The DBLEN parameter is superseded by the DBLOFFST parameter for HDAM and PHDAM SR by APAR PQ83931. The DBLEN parameter is supported only for the compatibility with the existing JCL streams.

System action: SRG continues processing.

User response: One of the following actions is required:

- If you ran SRG to generate an SR for a new HDAM or PHDAM database, run SRG again with the DBLOFFST parameter specified instead of the DBLEN parameter.
- If an SR exists for an HDAM or a PHDAM database, and if that SR was generated with SRG with the DBLEN parameter specified, if you run SRG to regenerate an SR for that database, check the number of RAPs per CI, which is printed as the second subparameter of the RMNAME parameter in the Recommended DBD Control Statement report. If the number is different from the value that was reported at the SRG run for the existing SR, you must unload and reload the database. Complete the following procedure:
  1. Unload the existing database by using the original SR.
  2. Run SRG with the DBLOFFST parameter specified.
  3. Reload the database by using the SR that was generated by SRG in the previous step.

If the reported number of RAPs per CI is the same as the one reported at the SRG run for the existing SR, you can use the generated SR without unloading and reloading the database.

FABO0131I THE NUMBER OF EQUATION TABLES IN THE SRT EXCEEDED TNUMLIM(, AREA : *xxxxxxxx***)**

Explanation: The number of equation tables in the SRT reached the maximum value that is specified by the TNUMLIM parameter during the equation table optimization. In the case of DEDB, *xxxxxxxx* shows the name of the DEDB area for which this message is issued.

System action: SRG ends the optimization process, and continues processing.

User response: None. This message is informational.

FABO0132I OPTIMIZATION ELAPSED TIME EXCEEDED OPTMTIME(, AREA : *xxxxxxxx***)**

Explanation: The elapsed time of the equation table optimization process reached the maximum time that is specified by the OPTMTIME parameter. In the case of DEDB, *xxxxxxxx* shows the name of the DEDB area for which this message is issued.

System action: SRG ends the optimization process, and continues processing.

User response: None. This message is informational.

FABO0133A THE TOTAL SR TABLE SIZE EXCEEDS 16 MB

Explanation: This message is issued when SRG determines that the total module size of the SRT exceeds 16 MB.

System action: SRG ends processing with RC=8.

User response: Take one of the following actions to prevent the module size from exceeding 16 MB:

- If SRTYPE=SINGLE is specified, specify SRTYPE=MULTI to divide the SRT into multiple modules.
- Reduce the number of keys in the input key file to reduce the number of equation tables in the SRT.

- Modify the optimization condition that is specified by the OPTMZ parameter to reduce the number of equation tables in the SRT.

FABO0134A THE SR TABLE SIZE FOR AREA *xxxxxxxx* EXCEEDS 16 MB

Explanation: This message is issued when SRG determines that the module size of the SRT for an area exceeds 16 MB. *xxxxxxxx* shows the name of the DEDB area.

System action: SRG ends processing with RC=8

User response: Take one of the following actions to prevent the module size from exceeding 16 MB:

- In the input key file, reduce the number of keys that belong to the indicated area to reduce the number of equation tables in the SRT.
- Modify the optimization condition that is specified by the OPTMZ parameter for the indicated area to reduce the number of equation tables in the SRT.

FABO0201I THE NUMBER OF DUPLICATE KEY IS *xxxxxxxx*

Explanation: One of the following occurred:

- Duplicate keys were found in the input key file. *xxxxxxxx* is the number of duplicate keys.
- When “O” was specified for the subkey field attribute (“c” in (a,b,c)) in the ANALYZE control statement, this message is issued also when the same compressed key was assigned to multiple input keys during key compression. In this case, these keys were not analyzed; therefore, they became synonyms.

System action: SRG processes only the first one of the duplicate keys, and ignores all others. Therefore, the number of input keys to be analyzed decreases by the number of the discarded keys. SRG continues processing, and eventually ends with RC=4.

User response: When “O” was specified for the subkey field attribute in the ANALYZE control statement, the total number of synonyms is the sum of the value shown in the SRG statistics report and the value *xxxxxxxx* in this message. Consider this matter when creating a database.

In other cases, take one of the following actions depending on the situation:

- When the existing duplicate keys may be ignored, use the results of the SRG execution.
- When there was an input key file creation error, re-create the input key file and run the SRG job again.
- When the parameter specifications on the ANALYZE control statement were wrong, correct the key attribute and the values of the INKEYL and INOFFST parameters according to the specifications of the input key file, and rerun the SRG job.

FABO0202I THE NUMBER OF ERROR ATTRIBUTE KEY IS *xxxxxxxx*

Explanation: The value of a key in the input data set did not match the attribute specified on the ANALYZE control statement as c of (a, b, c). For example, if (1, 4, N) is specified as the key attribute and 135A is entered as the input key, N and A does not match. *xxxxxxxx* is the number of erroneous keys.

System action: SRG ignores the erroneous input key data and continues processing. Therefore, the number of input keys to be analyzed decreases by the number of erroneous keys. SRG eventually ends with RC=4.

User response: Take one of the following actions depending on the situation:

- When the key may be ignored, use the results of the SRG execution.
- When there was an input key file generation error, regenerate the input key file and run the SRG job again.
- When the parameter specifications on the ANALYZE control statement were wrong, correct the key attribute and the values of the INKEYL and INOFFST parameters according to the specifications of the input key file, and rerun the SRG job.

FABO0203I THE NUMBER OF SEQUENCE ERRORS IS *xxxxxxxx*

Explanation: The keys in the input key file were not in ascending sequence. *xxxxxxxx* is the number of sequence error keys.

System action: SRG ignores the sequence error input key data and continues processing. Therefore, the number of input keys to be analyzed decreases by the number of error keys. SRG eventually ends with RC=4.

User response: Take one of the following actions depending on the situation:

- When the existing sequence error keys may be ignored, use the results of the SRG execution.
- When the input key file was not sorted in ascending sequence, sort the input key file in ascending sequence and rerun the SRG job.
- When the parameter specifications on the ANALYZE control statement were wrong, correct the key attribute and the values of the INKEYL and INOFFST parameters according to the specifications of the input key file, and rerun the SRG job.

FABO0204A NO VALID RECORD FOUND IN KEY FILE DDN = *xxxxxxxx*

Explanation: Although the input key file has been analyzed, there was no input key to be analyzed. *xxxxxxxx* is the ddname of the input key file.

System action: Subsequent processing depends on the database type specified on the ANALYZE control statement:

- DEDB

  SRG ignores processing of the present area and starts processing of the next area. If this message is issued to all the areas, SRG ends with RC=8. If the error occurred only for this area and processing of the other areas ends normally, SRG ends with RC=4.
- HDAM and PHDAM (VSAM or OSAM)

  SRG ends processing with RC=8.

User response: Take either of the following actions depending on the cause of the error:

- If any of the FABO0201I, FABO0202I, FABO0203I, or FABO0205I messages was issued before this message, refer to the Programmer Response of that message, take the necessary action, and rerun the SRG job.
- If none of the above messages was issued before this message, check and try one of the following:
  - When no record exists in the data set defined on the *xxxxxxxx* DD statement, create an input key file, specify the data set on the *xxxxxxxx* DD statement, and rerun the SRG job.
  - When no key exists in the range that is specified with the KEYRNG or ARETRY parameter of the AREADEF control statement, correct the value of the KEYRNG or ARETRY parameter or re-create the input key file, and then rerun the SRG job.

FABO0205I THE NUMBER OF DB RECORD SIZE ERRORS IS ***xxxxxxxx***

Explanation: The database record length of the input key data was shorter than the key length (INKEYL) that is specified on the ANALYZE control statement. This message is issued when the record length is specified with the DBLEN or DBLOFFST parameter. *xxxxxxxx* shows the total number of incorrect records.

System action: SRG ignores the incorrect key records and continues processing. SRG eventually ends processing with RC=4.

User response: Take one of the following actions depending on the error type:

- When the input key file contains an incorrect length record but it may be ignored, use the SRG execution results.
- When there was a creation error of the input key file, re-create the input key file so that the database record length is longer than the input key length, and rerun the SRG job.
- When the specification of the DBLEN or the DBLOFFST parameter of the ANALYZE control statement is incorrect, correct the specification of the DBLEN or DBLOFFST parameter according to the specification of the input key file, and rerun the SRG job.

Note: DBLEN parameter is accepted only to maintain compatibility. For HDAM and PHDAM databases, use DBLOFFST instead of DBLEN.

FABO0206A INVALID ATTRIBUTE FOUND IN [KEYRNG | ARETRY] PARAMETER

Explanation: The value of KEYRNG or ARETRY that is specified on the AREADEF control statement did not match the specification of the key attribute that is specified on the ANALYZE control statement. This message is issued, for example, when the following parameter values are specified on the control statement:

SRR	ANALYZE (1, 2, X)
AREA1	AREADEF KEYRNG = (0001, 00FQ)

In this example, the KEYRNG parameter must specify the key range in hexadecimal expressions. "Q," however, is not hexadecimal and causes an error.

System action: SRG ignores processing for the area that has the error, and continues processing for the other areas. It eventually ends with RC=4.

User response: Correct the KEYRNG or ARETRY parameter value of the AREADEF control statement so that it matches the key attribute that is specified on the ANALYZE control statement, and rerun the SRG job. See KEYRNG or ARETRY for the specification of the KEYRNG or ARETRY parameter.

FABO0207A OPTIMIZATION ERROR. TOO FEW RAPS ARE AVAILABLE. ***xxxxxxxx*** **RAPS REQUIRED**

Explanation: The number of RAPs is calculated from the database size specified on the control statement. This message is issued when the optimization conditions could not be met with this calculated number of RAPs. *xxxxxxxx* is the value obtained by multiplying the number of RAPs necessary for optimization by the maximum number of synonyms.

This value is calculated as shown below depending on the database type:

- DEDB (when UOW = (a,b) and ROOT=(c,d))
  ```
  xxxxxxxx = (a - b) x (c - d)
              x (Maximum Number of Synonyms + 1)
  ```
- HDAM and PHDAM (VSAM or OSAM)
  ```
  xxxxxxxx = (CIBYTE x CI Capacity Ratio / DBRECSZ
             x 100) x CITRK x TRCYL x PRIMCI
             x (Maximum Number of Synonyms + 1)
  ```

System action: Subsequent processing depends on the database type specified on the ANALYZE control statement:

- DEDB

  SRG ignores processing for the current area and starts processing of the next area. If this message is issued to all the areas, SRG ends with RC=8. If the error occurred only for this area and this area does not exist in SRRFILE and processing of the other areas ends normally, SRG ends with RC=4. If the error occurred only for this area and this area exists in SRRFILE and processing of the other areas ends normally, SRG ends with RC=0.

- HDAM and PHDAM (VSAM or OSAM)

  SRG ends processing with RC=8.

User response: Take one of the following actions depending on the database type:

- DEDB

  Increase the UOW and ROOT parameter values of the AREADEF control statement for the area that has the error, delete all the other AREADEF control statements, and rerun the SRG job.

  The necessary number of RAPs (N) is given by the following equation. When UOW = (a, b), and ROOT = (c, d), then:

  ```
  N = (a - b) x (c - d)
       x (Maximum Number of Synonyms + 1)
  N ≥ xxxxxxxx
  ```

- HDAM and PHDAM (VSAM or OSAM)

  Adjust each parameter and rerun the SRG job. The necessary number of RAPs (N) is given by the following equation:

  ```
  N = ((CIBYTE x Maximum CI Capacity Ratio)
       / (DBRECSZ x 100)) x CITRK x TRCYL
       x (Maximum Number of Synonyms + 1)
  N ≥ xxxxxxxx
  ```

FABO0208A [KEYRNG | ARETRY] VALUE IS NOT IN ASCENDING SEQUENCE

Explanation: The key range that is specified with the KEYRNG or ARETRY parameter of the AREADEF control statement is not in ascending sequence.

If KEYRNG=(100, 050) is specified, for example, the key range of this area is between LOW=100 and HIGH=050, and it does not make sense.

System action: SRG ignores processing for the area that has the error and starts processing of the next area. SRG eventually ends with RC=4.

User response: Correct the KEYRNG or ARETRY parameter value for the area that has the error so that the value is in ascending sequence and rerun the SRG job. When you rerun the job, you do not need to specify the AREADEF control statements for the areas that were processed normally during the first run.

FABO0209A TOO FEW VALID RECORDS FOUND IN KEY FILE DDN = ***xxxxxxxx***

Explanation: Although Option 0 was selected on the ANALYZE control statement, there were fewer than three keys in the input key file *xxxxxxxx*. The system requires three or more keys to create the equation table.

System action: Subsequent processing depends on the database type specified on the ANALYZE control statement:

- DEDB

  SRG ignores processing for the current area and starts processing the next area. If this message is issued to all the areas, SRG ends with RC=8. If the error occurred only for this area and processing of the other areas ends normally, SRG ends with RC=4.

- HDAM and PHDAM (VSAM or OSAM)

  SRG ends processing with RC=8.

User response: Change the OPTN parameter value of the ANALYZE control statement, or check to see if the contents of the input key file match the parameter value on the ANALYZE control statement, and rerun the SRG job.

FABO0210A THE STORAGE FOR COMPRESSED KEY FILE IS NOT ENOUGH TO CONTINUE PROCESSING, KEYNBR : ***yyyyyyyy*** **(, AREA :** ***xxxxxxxx*****)**

Explanation: The storage size of the compressed key file acquired by the KEYNBR parameter of the ANALYZE control statement was not enough. *yyyyyyyy* is the value specified in the KEYNBR parameter. *xxxxxxxx* is the name of the area that has the error, in the case of DEDB.

System action:

- HDAM and PHDAM

  SRG ends processing with RC=8.

- DEDB

  SRG ends processing for the area that has the error, and continues processing.

User response: Correct the KEYNBR parameter value, and rerun the SRG job.

FABO0211A NUMBER OF INPUT KEYS IS 16,777,216 OR GREATER

Explanation: The number of input keys in the key file is 16,777,216 or greater. The maximum number of non-duplicate input keys is 16,777,215.

System action: SRG ends processing with RC=8.

User response: Reduce the number of keys in the key file and rerun the SRG job.

FABO0212W THE NUMBER OF ARETRY KEY RANGES EXCEEDED THE ALLOWABLE MAXIMUM FOR AREA: ***xxxxxxxx*****. USE THE KEYRNG PARAMETER**

Explanation: The number of the ARETRY key ranges that is specified in the AREADEF control statement for the indicated area exceeds the maximum allowable number of characters.

The internal counter of the ARETRY key range table overflow and some key ranges are not used. As a result, the input keys that are within the discarded key ranges are not processed when SRG builds the equation table for the area and when SRG simulates randomizer calls for the area.

The maximum number of key ranges that can be specified on an ARETRY parameter is twice the number of bytes of actual key value multiplied by the number of ARETRY key range pairs, and the product of this equation must be less than 32759. Use the following formula to determine the maximum allowable number of ARETRY key range pairs:

```
(length of the key x 2 ) x (the number of ARETRY key range pairs ) < 32759
```

For example:

- If the length of the key is 5 bytes, up to 3000 key range pairs can be specified in an ARETRY parameter.
- If the length of the key is 25 bytes, up to 655 key range pairs can be specified in an ARETRY parameter.
- If the length of the key is 255 bytes, up to 64 key range pairs can be specified in an ARETRY parameter.

However, this upper limit is not checked by SRG. When the limit is exceeded, only a part of input keys is used for creating the equation table. As a result, the generated randomizer might not satisfy the specified optimization criteria for the CI capacity and the number of maximum synonyms. This message is a warning that this potential issue might exist.

System action: SRG continues processing, generates the SRT source member for the area, creates or updates the SRR source member with the correct master table entries for the area, and ends with RC=4.

Note: SRG does not stop processing for the compatibility with existing SRG JCLs and existing SRG DEDB randomizers.

User response: Unless you have a compatibility issue with the existing randomizer that has been generated in past jobs, consider replacing the ARETRY parameter with the KEYRNG parameter.

FABO1301A *xxxxxxxx* **: RECOMMENDED PARAMETER ROOT=(***x1***,***x2***),UOW=(***y1***,***y2***)**

Explanation: This message is issued when SRG has automatically increased the number of RAPs and changed either the ROOT or the UOW parameter. SRG makes these changes when, as a result of input key analysis or the optimization process, more RAPs become necessary than are specified in the control statement.

The initial total number of RAPs is calculated from the UOW and ROOT parameters of the AREADEF control statement, as follows:

```
number of RAPs = ( a - b) x (c - d),
```

where

a is the total number of CIs in the UOW

b is the number of CIs in the UOW overflow portion

c is the total number of UOWs in the area

d is the total number of UOWs in the overflow portion of the area

The values *x1*, *x2*, *y1*, and *y2* shown in the message text are the ROOT and UOW parameters that must be specified on the AREA statement of the DBDGEN for the DEDB area *xxxxxxxx* if you decide to use the generated randomizer.

System action: SRG continues processing.

User response: First, you must check the recommended DBDGEN parameters that are shown in the message.

If a value of the UOW or ROOT parameter has been changed, but you do not want to change the original value, take the following actions and rerun SRG:

- If the OPTMZ parameter is not specified, change the items such as the ROOT parameter of the AREADEF control statement so that the number of RAPs is satisfied.
- If the OPTMZ parameter is specified:
  - Relax the optimization conditions, especially the maximum CI capacity ratio.
  - Increase the size of the equation table, using the TSIZE parameter.

If the changed value is within the allowable range, use the DBDGEN parameters shown in the message for your DBDGEN.

FABO1401A SPACE OF RECORD OR DIRECTORY IS SHORTAGE. DDNAME = SRRFILE

Explanation: The SR source was stored temporarily in the ddname SRRFILE. This message is issued when there was space shortage either for record insertion or for member name registration in the directory while SRG was trying to store the SR source in the data set.

System action: Subsequent processing depends on the database type specified on the ANALYZE control statement:

- DEDB

  SRG ignores processing for the current area and starts processing the next area. If this message is issued to all the areas, SRG ends with RC=8. If the error occurred only for this area and processing of the other areas ends normally, SRG ends with RC=4.
- HDAM and PHDAM (VSAM or OSAM)

SRG ends processing with RC=8.

User response: Compress or reallocate the data set that was specified on the DD statement for SRRFILE. For HDAM and PHDAM, rerun SRG. For DEDB, regenerate the equation table for the area that has the error. This time, the AREADEF control statements for the normally-processed areas need not be specified.

FABO1402A SR IS NOT YET MIGRATED OR FORMAT OF SRR SOURCE IS INVALID, SR=*SR_name* **AREA=***AREA_name* **REASON=***x*

Explanation: An error was detected during the DEDB area processing of the area (*AREA_name*) in the case where the SRR source for the SR (*SR_name*) was already stored in the SR source library.

REASON=1
AREADEF control statement was specified, but the SR was not yet migrated into a new structure.

REASON=2
AREADEF control statement was specified, but the area specifications did not match between the control statement and the information stored in the SRR source.

System action: The SRT source for the DEDB area is not stored in the SR source library. SRG ends processing with RC=8.

User response:

REASON=1
Specify the ANALYZE control statement with SRCMIG=YES, LINKSR control statement with AREAID parameter, and rerun the job.

REASON=2
Check the control statement specifications, ensure that you use a correct SR source library (SRRFILE), and rerun the job.

FABO1501I SR LINK ERROR MEMBER = *xxxxxxxx* **NOT FOUND IN DDN = SRRFILE**

Explanation:

In case of HDAM and PHDAM:
When SR is generated, the SRR source and the SRT source are merged. This message is issued when the SRGEN=YES parameter was specified in the ANALYZE control statement, but the SRR source or SRT source did not exist in SRRFILE. *xxxxxxxx* is the missing SRR name or SRT name.

In case of DEDB:
When SR is generated, the SRR source and the equation tables for all the areas (areas whose name were specified with the AREAID parameter of the LINKSR control statement) are merged. This message is issued when the LINKSR control statement was specified, but the SRR source or DEDB area source did not exist in SRRFILE. *xxxxxxxx* is the missing SRR name or area name.

System action:

In case of HDAM and PHDAM:
SRG ends with RC=8.

In case of DEDB:
If no equation table exists in SRRFILE, SRG ends with RC=8. If one or more tables exist, SRG generates an SR load module with only the existing equation tables, and then ends with RC=4.

User response:

In case of HDAM and PHDAM:
If SRGEN=YES was specified in the ANALYZE control statement, even if the SRR source or SRT source did not exist in SRRFILE, do not specify SRGEN=YES and then rerun SRG. Otherwise, use the correct SRRFILE.

In case of DEDB:
Take one of the following actions depending on the error type:

- When the specification of AREAID on the LINKSR control statement was wrong, specify the correct area name with the AREAID parameter and rerun the SRG job.
- When no equation table was generated with the area name specified with the AREAID parameter of the LINKSR control statement, add the AREADEF control statement for the area and rerun the SRG job. This time, the AREADEF control statement for the normally processed areas need not be specified.

FABO1502A SR IS NOT YET MIGRATED OR INVALID FORMAT, LINKSR IS CANCELED FOR SR=*SR_name*

Explanation: LINKSR processing with SRCMIG=NO (default) was specified, but the SR (*SR_name*) was not yet migrated into a new structure.

System action: SRG does not generate the SR load module, and ends processing with RC=8.

User response: Specify ANALYZE control statement with SRCMIG=YES, and LINKSR control statement with AREAID parameter. Rerun the SRG job.

FABO1503A SRR IS MIGRATED, BUT AREA IS NOT MIGRATED FOR SR=*SR_name* **AREA=***AREA_name*

Explanation: SRR source was already migrated, but the SRT source for the DEDB area (*AREA_name*) was not yet migrated for the SR (*SR_name*). This message is issued when the area name was not specified in the AREAID parameter when migration was specified (SRCMIG=YES in the ANALYZE control statement).

System action: SRG does not generate the load module, and ends with RC=8.

User response: Specify the ANALYZE control statement with SRCMIG=YES, LINKSR control statement with the area name in the AREAID parameter, and rerun SRG.

FABO1504A SR IS MIGRATED, BUT AREA FORMAT IS INVALID FOR SR=*SR_name* **AREA=***AREA_name*

Explanation: SR (*SR_name*) was already migrated, but the SRT source for the DEDB area (*AREA_name*) was not in a correct structure.

System action: SRG does not generate the load module, and ends with RC=8.

User response: This DEDB area source is not usable. Create a new DEDB area source by specifying ANALYZE control statement and AREADEF control statement.

FABO1505W AREA IS NOT INCLUDED INTO SR, FOR SR=*SR_name* **AREA=***AREA_name*

Explanation: SR source for the SR (*SR_name*) contained an SRT source for the DEDB area (*AREA_name*) that was not specified in the AREAID parameter.

System action: SRG generates an SR load module ignoring this area, and ends with RC=4.

User response: If this DEDB area is not used, ignore this message. If the DEDB area is used, add this area name in the AREAID parameter, specify only this DEDB area in the LINKID parameter, and rerun SRG.

FABO1506W SLOPL PARAMETER IN *mmmm* **STATEMENT IS CHANGED FROM** *xxx* **TO** *yyy*

Explanation: The value for the SLOPL parameter of the macro statement '*mmmm*' was changed from '*xxx*' to '*yyy*', because the SLOPL value '*xxx*' set in the statement was not the maximum of all SRT SLOPL values.

System action: SRG continues its processing.

User response: If you can assume that the correct SRR and SRT source files were used, you can ignore this warning message.

FABO1801I SOURCE MIGRATION COMPLETED FOR SR=*SR_name*

Explanation: SRCMIG=YES was specified in the ANALYZE control statement, and the SR source for the SR (*SR_name*) was migrated to a new structure successfully.

System action: SRG continues processing to generate an SR load module.

User response: None. This message is informational.

FABO1802W AREA SOURCE IS ALREADY MIGRATED FOR SR=*SR_name* **AREA=***AREA_name*

Explanation: SRCMIG=YES was specified in the ANALYZE control statement, but the SRT source for the DEDB area (*AREA_name*) was already migrated to a new structure for the SR (*SR_name*).

System action: SRG ends the migration for the area, and continues processing.

User response: If the message FABO1801I is issued after this message, ignore this message. Otherwise, refer to the message FABO1803A.

FABO1803A SRCMIG OR SRCCHK FAILED FOR SR=*sr_name* **MEMBER=***AREA_name* **REASON=***x*

Explanation: The source for the SR (*sr_name*) could not be migrated, or the source format checking has failed, for one of the following reasons:

Reason (*x*)	Description
1	The SRRFILE did not contain an SRT source file for the DEDB area (*AREA_name*), or an SRR source file.
2	The format of the SRR source file was incorrect. An invalid statement was found, or the statement sequence was incorrect.
3	The format of the SRT source file for the DEDB area (*AREA_name*) was incorrect. An invalid statement was found, or the statement sequence was incorrect.
4	The format of the SRR source file was incorrect. An XPRI record was found even though the SRR source file had not been migrated to the new format.
5	An XPRI macro statement for the DEDB area that was not specified on the AREAID parameter of the LINKSR control statement was found in the SRR source file.
6	A macro statement is missing from the SRR source file, or the SDOLVL record is missing from the SRR source file.

7 The XPRS macro statement was found in the SRT source file for the DEDB area (*AREA_name*) although the SRR source file had the old format.

8 A macro statement is missing from the SRT source file for the DEDB area (*AREA_name*).

9 The number of records for an XPRI macro statement is invalid for the DEDB area, *AREA_name*, or no XPRI statement is found for the area.

If SRCMIG=YES is specified, this message is issued when the SRG encounters the error while processing the XPRI macro statements in the area, *AREA_name*.

If SRCCHK=1 is specified, this message is issued while SRG is processing the XPRI records for the area, *AREA_name*, in the SRR source file.

If SRCCHK=2 is specified, this message is issued while SRG is processing the XPRI records in the SRT source file of the area, *AREA_name*.

System action: SRG stops its processing and ends with RC=8. If SRCMIG=YES is specified, the migration statuses are as follows:

Reason (*x*)	Description
1	If this message is issued for the SRR source file, the SRR source file and all SRT source files are not migrated to the new format. If this message is issued for an SRT source file, the SRR source file is migrated to the new format with the pending status set, but no SRT source file is migrated to the new format.
2, 4, 5	The SRR source file and all SRT source files are not migrated to the new format.
3, 7, 8, 9	The SRR source file is migrated to the new format with the pending status set, but no SRT source file is migrated to the new format.

User response: The following response is required:

Reason (*x*)	Description
1	Specify the correct SR name on the ANALYZE statement and the correct area names on the AREAID parameter of LINKSR control statement. Rerun the job.
2, 4, 5	Make sure that the correct SRR source file and the correct SRRFILE library are being used. Rerun the job.
3, 7, 8	Make sure that the correct SRT source files and the correct SRRFILE library are being used. Rerun the job.
6	Make sure that the correct SRR source file and the correct SRRFILE library are being used. If the correct SRR source file is being used, but it has not yet migrated to the new format, run SRG, specifying SRCMIG=YES.
9	If SRCMIG=YES is specified and this message is issued, make sure that the correct SRT source file for the indicated area is being used, and rerun the job. If SRCCHK=1 is specified and this message is issued, specify SRCCHK=2 and rerun the job. If SRCCHK=2 is specified and this message is issued, check whether the correct SRT source file for the indicated area is being used.

FABO1804I XPRI RECORDS FOR AREA=*AREA_name* OF SR=*sr_name* ARE REBUILT FROM SRT

Explanation: This message notifies you that the XPRI macro statements in the SRR source file for the DEDB area (*AREA_name*) have been rebuilt from the SRT source file of the area because the XPRI statements were missing or in an incorrect format. This message is issued only when SRCCHK=2 is specified on the LINKSR control statement, and is preceded by the message FABO1805W.

System action: SRG continues its processing.

User response: Verify that this message has been issued for each DEDB area for which the message FABO1805W was issued. See also the Programmer Response for message FABO1805W.

FABO1805W INCORRECT XPRI FOMRAT OR XPRI STATEMENTS ARE MISSING FOR AREA=*AREA_name* OF SR=*sr_name*

Explanation: This message is issued only when SRCCHK=2 is specified on the LINKSR control statement. It notifies you that the XPRI statements for the DEDB area named are missing or in an incorrect format. This message should be followed by the message FABO1804I.

System action: SRG continues processing and ends with RC=4.

User response: If you can assume that the correct SRR and SRT source files have been used, and you have verified that the message FABO1804I was issued for the indicated DEDB area, you can ignore this warning message. If you used the wrong SRR source file or SRT source files, prepare the correct ones and rerun the job with SRCCHK=1.

FABO1806W SLOPL PARAMETER IN *mmmm* STATEMENT IS CHANGED FROM *xxx* TO *yyy*

Explanation: The value for the SLOPL parameter of the macro statement '*mmmm*' has been changed from '*xxx*' to '*yyy*', because the SLOPL value '*xxx*' set in the statement was not the maximum of all SRT SLOPL values.

System action: SRG continues processing and ends with RC=4.

User response: If you can assume that the correct SRR and SRT source files have been used, you can ignore this warning message.

FABO1901A ORIGINAL SOURCE OF *SR_name* NOT FOUND ALTHOUGH TYPECHG=YES

Explanation: This message is issued when the original SR source file *SR_name* does not exist in the source file library SRRFILE although TYPECHG=YES is specified in the ANALYZE statement.

System action: SRG does not process the type change function and ends processing with RC=8.

User response: Specify TYPECHG=NO or delete the TYPECHG parameter in the ANALYZE statement, and rerun the SRG job.

FABO1902A TYPECHG FUNCTION NEEDS MIGRATED SR SOUCE

Explanation: The original SR source was generated by the IAPP version of SDO and has not been migrated to the new structure, although TYPECHG=YES is specified. The TYPECHG function always needs the original source with its SRT of multiple CSECT type.

System action: SRG does not process the type change function and ends the processing with RC=8.

User response: Specify SRCMIG=YES in the ANALYZE statement, and rerun the SRG job. If the original SR is generated by IAPP Version 1 SDO, then observe the migration procedure described in the note in Chapter 8, "Reference: Migration procedures," on page 123.

FABO1903A SRTYPE IN ANALYZE STATEMENT NOT MATCH ALTHOUGH TYPECHG=NO

Explanation: The value of SRTYPE= in the ANALYZE statement does not match the SR type of the existing original SR, although TYPECHG=NO.

System action: SRG ends with RC=8.

User response: If you want to change the SR types, then specify TYPECHG=YES in the ANALYZE statement and rerun the SRG job. If you do not want to change SR types, then specify the same type to SRTYPE= parameter on the ANALYZE statement as the type of the existing SR, and rerun the SRG job.

FABO1904A CANNOT CHANGE TYPES, SINCE THE SR IS ALREADY OF TYPE *SR_type*

Explanation: TYPECHG=YES and SRTYPE=*SR_type* are specified in the ANALYZE statement, but the type of the existing SR is already *SR_type*. This message could also be issued, when the extended master table is not found in IMSLIB although the original SR source in SRRFILE is the one for a multiple type SR.

System action: SRG does not process the type change function and ends processing with RC=8.

User response:

- If you do not want to change the type of the SR, specify TYPECHG=NO or delete the TYPECHG parameter. Then, rerun the SRG job.
- If you want to change the type of the SR, make sure that the current SR module type is not the one that is specified in the ANALYZE statement. If recovery of the SR load modules is needed, make the recovery. Then rerun the SRG job.

FABO1905A MODULE *mod_name* NOT FOUND ALTHOUGH TYPECHG=YES AND SRTYPE=*SR_type*

Explanation: This message is issued when the module *mod_name* is not found in IMSLIB although it should exist in order to change the module type of the DEDB SR to *SR_type*. This message could also be issued when the user specified TYPECHG=YES for the DEDB SR, which is generated by IAPP Version 1 SDO, has not been migrated to the new structure.

System action: SRG does not process the type change function and ends processing with RC=8.

User response:

- If you do not want to change the type of the SR, specify TYPECHG=NO or delete the TYPECHG parameter. Then, rerun the SRG job.
- If you want to change the type of the SR, make sure that the current SR module type is not the one that is specified in the ANALYZE statement. If recovery of the SR load modules is needed, make the recovery. Then rerun the SRG job.
- If the exiting SR is the one that was generated by IAPP V1 SDO and has not been migrated, observe the migration procedure for this type of SR. See the note in Chapter 8, "Reference: Migration procedures," on page 123.

FABO1906A MODULE *xxxxxxxx* IS NOT FOUND IN IMSLIB THOUGH LINKID=ALL IS NOT SPECIFIED

Explanation: This message is issued when some (but not all) areas were selected in the LINKID parameter or the SR name was specified in the LINKID parameter, but the module *xxxxxxxx* is not found in IMSLIB although it should exist in the library for the LINKSR processing for the SR.

System action: LINKSR processing is canceled. SRG does not generate the SR load module, and ends the job with RC=8.

User response: Make sure that all modules necessary for the SR exist in the load module library. If the SR load module library (IMSLIB) is not specified correctly, run SRG again with the correct IMSLIB specification.

If the SR load module library is specified correctly;

- If the module *xxxxxxxx* does not exist in the library, but all source files for the SR exist in the SR source library, then specify LINKID=ALL in the LINKSR control statement or add the name *xxxxxxxx* to the LINKID parameter.
- If the module *xxxxxxxx* does not exist in the library and the source file of *xxxxxxxx* does not exist in the SR source library, specify an appropriate AREADEF control statement to generate the source file of *xxxxxxxx* and specify LINKID=ALL in the LINKSR control statement or add the name *xxxxxxxx* to the LINKID parameter.

Then, rerun the SRG job.

FABO3999A UNKNOWN ERROR OCCURRED IN *modulename* MODULE RC = *nn*

Explanation: SRG found that an unknown error occurred in the SRG internal module (*modulename*). This kind of error should not occur. RC (*nn*) is the internal reason code.

System action: SRG ends with an abend code of 3999.

User response: Report the problem to IBM.

FABO8000A ATTRIB=REUS/RENT IS NOT SUPPORTED IN THIS IMS RELEASE

Explanation: This message is issued if you run SRG in an IMS environment earlier than IMS Version 7 with SRGEN=YES specified on the ANALYZE control statement, and if ATTRIB=REUS or =RENT is specified for the SR. This message is printed in the assemble listing of the SR. A reusable (ATTRIB=REUS) or reentrant (ATTRIB=RENT) SR is not supported by SRG for any IMS environment earlier than IMS Version 7.

System action: SRG processing ends with RC=8.

User response: Remove the ATTRIB parameter from the ANALYZE control statement and rerun the SRG job to generate an SR with the nonreusable attribute. If you are to use the SR in IMS Version 7 or later, you must specify the appropriate IMS macro library on MACLIB DD for your SRG job.

SRG control statement messages

The following messages are issued in the SRG control statement assembly listing (ddname: SYSPRINT), when the SRG control statements are analyzed by SRG.

All the messages have the same message number (FABO9000A) and a submessage number (such as ANALYZE-001) to identify each message. Message FABO9000W (ANALYZE-022) is the only exception.

Note: The following aliases are provided for the compatibility with DBT Version 2 SDO. The following table also shows the name of macro that corresponds to each name contained in the submessage number.

Table 15. Macro names that correspond to each name contained in the submessage number

Name in submessage number	Macro name	Macro alias
XALPHACK	FABOALCK	-
XALPHA	FABOALPH	-
XPRJALST	FABOALST	-
ANALYZE	FABOANA	ANALYZE
XARCHK	FABOARCK	-
XARCVT	FABOARCV	-
AREADEF	FABOARE	AREADEF
XCOMP	FABOC	XCOMP
XCONBIN	FABOCBIN	-
XCEXT	FABOCEXT	-
XCOMPC	FABOCMPC	-
XDSE	FABODSE	XDSE
XDEF	FABOF	XDEF
XDEG	FABOG	XDEG
XHEX	FABOHEX	-
XPRI	FABOI	XPRI
XPRJ	FABOJ	XPRJ
XKANA	FABOKANA	-
XKANACH	FABOKNCH	-
LINKSR	FABOLK	LINKSR
XPRM	FABOM	XPRM
XMULT	FABOMULT	-
XNUMBER	FABONMBR	-
XNUMCK	FABONMCK	-
XNOTE	FABONOTE	-
XPACKED	FABOPKED	-
XPACKSS	FABOPKSS	-
XQU	FABOQU	-
XRANDOM	FABORND	-
XRANDT	FABORNDT	-

Table 15. Macro names that correspond to each name contained in the submessage number (continued)

Name in submessage number	Macro name	Macro alias
XPRS	FABOS	XPRS
XSUB	FABOSUB	-
XSDOTBL	FABOTBL	-
XPRX	FABOX	XPRX

FABO9000A (ANALYZE-001) ** ANALYZE CONTROL STATEMENT DUPLICATE

Explanation: Two or more ANALYZE control statements were specified. Only one ANALYZE control statement is allowed for each SRG job.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Delete the unnecessary ANALYZE control statement and rerun the SRG job.

FABO9000A (ANALYZE-002) ** ILLEGAL OPTMZ PARAMETER

Explanation: The OPTMZ parameter specification on the ANALYZE control statement is incorrect.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the OPTMZ parameter specification and rerun the SRG job. For specification of the OPTMZ parameter, see OPTMZ in "For SR and equation table generation method specifications" on page 40.

FABO9000A (ANALYZE-003) ** INBLKSZ IS NOT MULTIPLE OF INLRECL

Explanation: The INBLKSZ parameter value that specifies the block size of the input key file to be used by SRG was not an integral multiple of the INLRECL parameter value that specifies the record length. It must be an integral multiple of the record length.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the values of INBLKSZ and INLRECL and rerun the SRG job.

FABO9000A (ANALYZE-004) ** INPUT KEY FIELD NOT WITHIN INLRECL

Explanation: The INOFFST parameter value that indicates the offset of the key field within the input key file record is incorrect. This message is issued, for example, for the following specification:

INKEYL=4, INOFFST=5, and INLRECL=7

In the above example, the key field is too large to fit in the record.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the values of INKEYL, INOFFST, and INLRECL on the ANALYZE control statement so that the key field fits into a record. Then, rerun the SRG job.

FABO9000A (ANALYZE-008) ** NOAREA PARAMETER IS INVALID IN HDAM

Explanation: Both TYPE=VSAM or OSAM and the NOAREA parameter were specified on the ANALYZE control statement. The NOAREA parameter can be specified only when TYPE=DEDB.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the NOAREA parameter was specified for the database type HDAM or PHDAM (OSAM or VSAM), delete the NOAREA parameter and rerun the SRG job.
- When the database type is DEDB, specify TYPE=DEDB and rerun the SRG job.

FABO9000A (ANALYZE-009) ** CORE PARAMETER IS INVALID IN DEDB

Explanation: Both TYPE=DEDB and the CORE parameter were specified on the ANALYZE control statement. The CORE parameter can be specified only for the database type HDAM or PHDAM (TYPE=OSAM or VSAM).

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the database type is DEDB and the CORE parameter is specified, delete the CORE parameter and rerun the SRG job.

- When the database type is HDAM or PHDAM (OSAM or VSAM), specify TYPE=OSAM or VSAM and rerun the SRG job.

FABO9000A (ANALYZE-010) ** CIBYTE PARAMETER IS INVALID IN DEDB

Explanation: Both TYPE=DEDB and the CIBYTE parameter were specified on the ANALYZE control statement. The CIBYTE parameter can be specified only when TYPE=OSAM or VSAM.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the database type is DEDB and the CIBYTE parameter is specified, delete the CIBYTE parameter and rerun the SRG job.
- When the database type is HDAM or PHDAM (TYPE=OSAM or VSAM), specify TYPE=OSAM or VSAM and rerun the SRG job.

FABO9000A (ANALYZE-011) ** CITRK PARAMETER IS INVALID IN DEDB

Explanation: Both TYPE=DEDB and the CITRK parameter were specified on the ANALYZE control statement. The CITRK parameter can be specified only when TYPE=OSAM or VSAM.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the database type is DEDB and the CITRK parameter is specified, delete the CITRK parameter and rerun the SRG job.
- When the database type is HDAM or PHDAM (TYPE=OSAM or VSAM), specify TYPE=OSAM or VSAM and rerun the SRG job.

FABO9000A (ANALYZE-012) ** TRCYL PARAMETER IS INVALID IN DEDB

Explanation: Both TYPE=DEDB and the TRCYL parameter were specified on the ANALYZE control statement. The TRCYL parameter can be specified only when TYPE=OSAM or VSAM.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the database type is DEDB and the TRCYL parameter is specified, delete the TRCYL parameter and rerun the SRG job.
- When the database type is HDAM or PHDAM (TYPE=OSAM or VSAM), specify TYPE=OSAM or VSAM and rerun the SRG job.

FABO9000A (ANALYZE-013) ** PRIMCI PARAMETER IS INVALID IN DEDB

Explanation: Both TYPE=DEDB and the PRIMCI parameter were specified on the ANALYZE control statement. The PRIMCI parameter can be specified only when TYPE=OSAM or VSAM.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the database type is DEDB and the PRIMCI parameter is specified, delete the PRIMCI parameter and rerun the SRG job.
- When the database type is HDAM or PHDAM M (TYPE=OSAM or VSAM), specify TYPE=OSAM or VSAM and rerun the SRG job.

FABO9000A (ANALYZE-014) ** DBRECSZ PARAMETER IS INVALID IN DEDB

Explanation: Both TYPE=DEDB and the DBRECSZ parameter were specified on the ANALYZE control statement. The DBRECSZ parameter can be specified only when TYPE=OSAM or VSAM.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the database type is DEDB and the DBRECSZ parameter is specified, delete the DBRECSZ parameter and rerun the SRG job.
- When the database type is HDAM or PHDAM (TYPE=OSAM or VSAM), specify TYPE=OSAM or VSAM and rerun the SRG job.

FABO9000A (ANALYZE-015) ** FRECI PARAMETER IS INVALID IN DEDB

Explanation: Both TYPE=DEDB and the FRECI parameter were specified on the ANALYZE control statement. The FRECI parameter can be specified only when TYPE=OSAM or VSAM.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the database type is DEDB and the FRECI parameter is specified, delete the FRECI parameter and rerun the SRG job.

- When the database type is HDAM or PHDAM (TYPE=OSAM or VSAM), specify TYPE=OSAM or VSAM and rerun the SRG job.

FABO9000A (ANALYZE-017) ** NO SPECIFICATION OF DBLOFFST AND DBRECSZ

Explanation: TYPE=OSAM or TYPE=VSAM was specified, but neither the DBLOFFST nor the DBRECSZ parameter was specified.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the correct DBLOFFST and DBRECSZ parameters and rerun the SRG job.

FABO9000A (ANALYZE-018) ** DBLEN FIELD OVERLAPS KEY FIELD

Explanation: The field that contains the database record length overlaps the key field in the input database key record.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the correct DBLEN parameter and rerun the SRG job.

FABO9000A (ANALYZE-019) ** DBLOFFST FIELD OVERLAPS KEY FIELD

Explanation: The field that contains the database record length overlaps the key field in the input database key record.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the correct DBLOFFST parameter and rerun the SRG job.

FABO9000A (ANALYZE-020) ** DBLOFFST PARAMETER IS INVALID IN DEDB

Explanation: Both TYPE=DEDB and the DBLOFFST parameter were specified on the ANALYZE control statement. The DBLOFFST parameter can be specified only when TYPE=OSAM or TYPE=VSAM is specified.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the database type:

- When the database type is DEDB and if the DBLOFFST parameter is specified, delete the DBLOFFST parameter and rerun the SRG job.
- When the database type is HDAM or PHDAM (that is, TYPE=OSAM or TYPE=VSAM), specify TYPE=OSAM or VSAM and rerun the SRG job.

FABO9000A (ANALYZE-021) ** DBLOFFST PARAMETER CANNOT BE SPECIFIED WITH DBLEN PARAMETER

Explanation: Both the DBLEN and DBLOFFST parameters were specified on the ANALYZE control statement. Either the DBLEN parameter or the DBLOFFST parameter can be specified.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: The DBLEN parameter is superseded by the DBLOFFST parameter. Take one of the following actions depending on the error type:

- If you are using the DBLEN parameter for compatibility, delete the DBLOFFST parameter and rerun the SRG job.
- If you are using SRG to generate a new HDAM or PHDAM randomizer, delete the DBLEN parameter and rerun the SRG job.
- If you are using the DBLOFFST parameter instead of the DBLEN parameter, complete the following steps to migrate the SR:
  1. Unload the existing database by using the original SR.
  2. Run SRG by specifying the DBLOFFST parameter.
  3. Reload the database by using the SR that is generated by SRG in the previous step.

FABO9000A (ANALYZE-022) ** DBLEN PARAMETER IS OBSOLETE

Explanation: The DBLEN parameter is superseded by the DBLOFFST parameter.

System action: SRG continues processing.

User response: See the User response section of the FABO0130W message.

FABO9000A (ANALYZE-023) ** CALLTYPE IS INVALID IN TYPE=HDAM

Explanation: Both TYPE=VSAM or TYPE=OSAM and the CALLTYPE parameter were specified on the ANALYZE control statement. The CALLTYPE parameter can be specified only when TYPE=DEDB.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- If the CALLTYPE parameter was specified for the database type HDAM or PHDAM (that is, TYPE=OSAM or TYPE=VSAM), delete the CALLTYPE parameter and rerun the SRG job.
- If the database type is DEDB, specify TYPE=DEDB and rerun the SRG job.

FABO9000A (ANALYZE-101) ** INVALID ERPROC PARAMETER

Explanation: The ERPROC parameter of the ANALYZE control statement was incorrect.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the ERPROC parameter and rerun SRG.

FABO9000A (ANALYZE-103) ** INVALID SRGEN PARAMETER

Explanation: The SRGEN parameter of the ANALYZE control statement was incorrect.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-104) ** SRGEN IS INVALID IN TYPE=DEDB

Explanation: TYPE=DEDB and the SRGEN parameter were specified on the ANALYZE control statement at the same time. The SRGEN parameter can be specified only when TYPE=OSAM or TYPE=VSAM.

System action: SRG issues the FABO0107A message and the SRG job ends with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-105) ** INVALID DBM PARAMETER

Explanation: An error was detected in the DBM parameter of the ANALYZE control statement.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-106) ** DBM IS INVALID IN TYPE=HDAM

Explanation: TYPE=OSAM (or VSAM) and the DBM parameter were specified on the ANALYZE control statement at the same time. The DBM parameter can be specified only when TYPE=DEDB.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-107) ** INVALID KEYNBR PARAMETER

Explanation: An error was detected in the KEYNBR parameter of the ANALYZE control statement. Or, a KEYNBR parameter was specified while the subkey field attribute was 'O.' You cannot specify the KEYNBR parameter for generating a nonsequential randomizing module.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-108) ** INVALID TLIMIT PARAMETER

Explanation: An error was detected in the TLIMIT parameter of the ANALYZE control statement. Or, the TLIMIT parameter was specified without the OPTMZ parameter.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-109) ** TLIMIT IS INVALID IN TYPE=DEDB

Explanation: TYPE=DEDB and the TLIMIT parameter were specified on the ANALYZE control statement at the same time. The TLIMIT parameter can be specified only when TYPE=OSAM or TYPE=VSAM.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-110) ** INVALID SRCMIG PARAMETER

Explanation: The value specified in the SRCMIG parameter of the ANALYZE control statement is not correct.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-111) ** SRCMIG IS INVALID IN TYPE=HDAM

Explanation: The SRCMIG parameter was specified for TYPE=OSAM or TYPE=VSAM database. The SRCMIG parameter can be specified only for TYPE=DEDB database.

System action: SRG issues FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-112) ** INVALID SRTYPE PARAMETER

Explanation: The value specified in the SRTYPE parameter of the ANALYZE control statement is not correct.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-113) ** SRTYPE IS INVALID IN TYPE=HDAM

Explanation: The SRTYPE parameter was specified for TYPE=OSAM or TYPE=VSAM database. The SRTYPE parameter can be specified only for TYPE=DEDB database.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-114) ** INVALID TYPECHG PARAMETER

Explanation: The value specified in the TYPECHG parameter of the ANALYZE control statement is not correct.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-115) ** TYPECHG IS INVALID IN TYPE=HDAM

Explanation: The TYPECHG parameter was specified for TYPE=OSAM or TYPE=VSAM database. The TYPECHG parameter can be specified only for TYPE=DEDB database.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-116) ** INVALID NAME SPECIFIED ON AREANM

Explanation: The name specified in the AREANM parameter of the ANALYZE control statement is not allowed for the SR to be generated.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: See the explanation of the AREANM parameter and specify the correct parameter. Then, rerun the SRG job.

FABO9000A (ANALYZE-117) ** INVALID LNKPGM PARAMETER

Explanation: The LNKPGM parameter of the ANALYZE control statement was incorrect.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the LNKPGM parameter and rerun SRG.

FABO9000A (ANALYZE-118) ** INVALID CALLTYPE PARAMETER

Explanation: The value specified in the CALLTYPE parameter of the ANALYZE control statement is not correct.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-119) ** CALLTYPE=XCI IS INVALID FOR THIS IMS RELEASE

Explanation: The XCI (Extended Call Interface) for the DEDB randomizer is not supported by the IMS which you were to use in the SRG run.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-120) ** CALLTYPE=XCI IS INVALID WHEN SRTYPE=SINGLE

Explanation: The CALLTYPE=XCI is specified with SRTYPE=SINGLE. The CALLTYPE=XCI is allowed only for the randomizer of SRTYPE=MULTI.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Remove the CALLTYPE parameter or specify CALLTYPE=STD and rerun the SRG job.

FABO9000A (ANALYZE-121) ** INVALID ATTRIB PARAMETER

Explanation: A value other than blank, REUS, or RENT is specified for the ATTRIB parameter of the ANALYZE control statement.

System action: SRG processing ends with RC=8.

User response: Specify the keyword REUS or RENT for the ATTRIB parameter; or, remove the ATTRIB parameter.

FABO9000A (ANALYZE-122) ** ATTRIB PARAMETER IS INVALID IN THIS IMS RELEASE

Explanation: The parameter ATTRIB=REUS or ATTRIB=RENT is specified in an IMS environment earlier than IMS Version 7. ATTRIB=REUS and ATTRIB=RENT are valid only in IMS Version 7 or later.

System action: SRG processing ends with RC=8.

User response: Remove the ATTRIB parameter from the ANALYZE control statement, and rerun the SRG job to generate an SR with the nonreusable attribute. If you are to use the SR in IMS Version 7 or later, you must specify the appropriate IMS macro library on MACLIB DD for your SRG job.

FABO9000A (ANALYZE-123) ** ATTRIB PARAMETER IS INVALID FOR DEDB SR

Explanation: A non-blank value is specified for the ATTRIB parameter for DEDB SR. A non-blank ATTRIB parameter cannot be specified for DEDB SRs. DEDB SRs are always generated as reentrant modules.

System action: SRG processing ends with RC=8.

User response: Remove the ATTRIB parameter.

FABO9000A (ANALYZE-124) ** INVALID TNUMLIM PARAMETER

Explanation: Either an error was detected in the TNUMLIM parameter of the ANALYZE control statement, or the TNUMLIM parameter was specified without an OPTMZ parameter.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Correct the parameters and rerun SRG.

FABO9000A (ANALYZE-125) ** TNUMLIM IS INVALID WHEN TYPE=DEDB

Explanation: TYPE=DEDB and the TNUMLIM parameter were specified on the ANALYZE control statement. The TNUMLIM parameter can be specified only when TYPE=OSAM or TYPE=VSAM.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Correct the parameters and rerun SRG.

FABO9000A (ANALYZE-126) ** INVALID OPTMTIME PARAMETER

Explanation: An error was detected in the OPTMTIME parameter of the ANALYZE control statement, or the OPTMTIME parameter was specified without an OPTMZ parameter.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Correct the parameters and rerun SRG.

FABO9000A (ANALYZE-127) ** OPTMTIME IS INVALID WHEN TYPE=DEDB

Explanation: TYPE=DEDB and the OPTMTIME parameter were specified on the ANALYZE control statement. The OPTMTIME parameter can be specified only when TYPE=OSAM or TYPE=VSAM is specified.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-128) ** INVALID SIZECHK PARAMETER

Explanation: An error was detected in the SIZECHK parameter of the ANALYZE control statement.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify the correct parameter and rerun SRG.

FABO9000A (ANALYZE-999) ** INVALID PARAMETER: *xxxxxxxx*

Explanation: An incorrect parameter was specified on the ANALYZE control statement. *xxxxxxxx* is the incorrect parameter specified.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: *xxxxxxxx* cannot be specified. Correct the parameter and rerun SRG.

FABO9000A (AREADEF-001) ** ANALYZE CONTROL STATEMENT MISSING

Explanation: No ANALYZE control statement preceded the AREADEF control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify an ANALYZE control statement before the AREADEF control statement and rerun the SRG job.

FABO9000A (AREADEF-002) ** IGNORE BECAUSE TYPE IS *xxxx*

Explanation: The AREADEF control statement was specified, but the database type (specified by the TYPE parameter of the ANALYZE control statement) was not DEDB. The AREADEF control statement can be specified only when TYPE=DEDB. xxxx indicates the database type specified with the TYPE parameter of the ANALYZE control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the AREADEF control statement is specified for the database type HDAM or PHDAM (VSAM or OSAM), delete the AREADEF control statement and rerun the SRG job.
- When database type specification on the ANALYZE control statement is incorrect, correct the database type specification to TYPE=DEDB and rerun the SRG job.

FABO9000A (AREADEF-003) ** INVALID UOW PARAMETER

Explanation: The UOW parameter specification on the AREADEF control statement is not correct. It must be specified as shown below:

```
UOW = (n₁, o₁)
  n₁:  Total number of CIs within a UOW
  o₁:  Number of CIs in the overflow area
                           of a UOW.
```

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the UOW parameter specification and rerun the SRG job.

FABO9000A (AREADEF-004) ** INVALID ROOT PARAMETER

Explanation: The ROOT parameter specification on the AREADEF control statement is not correct. It must be specified as shown below:

```
ROOT = (n₂, o₂)
  n₂:  Total number UOWs per DEDB area
  o₂:  Number of UOWs in the overflow area of
                          the DEDB area.
```

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the ROOT parameter specification and rerun the SRG job.

FABO9000A (AREADEF-005) ** NO AVRECL PARAMETER

Explanation: Neither the DBLEN parameter of the ANALYZE control statement nor the AVRECL parameter of the AREADEF control statement was specified. When SR is optimized and simulated, the database record length is also checked; therefore, its information must be given with either of the above parameters.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- If a two-byte field containing the database record length exists in the input key field, set the position of the two-byte field within the input key file record, in the DBLEN parameter of the ANALYZE control statement.
- If the database record length information does not exist in the input key file, set the average database record length in the AVRECL parameter of the AREADEF control statement.

FABO9000A (AREADEF-006) ** INVALID DELCHAR PARAMETER

Explanation: More than one character was specified for the DELCHAR parameter of the AREADEF control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify only one character for the DELCHAR parameter and rerun the SRG job.

FABO9000A (AREADEF-007) ** INVALID CVCHAR PARAMETER

Explanation: More than one character was specified for the CVCHAR parameter of the AREADEF control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify only one character for the CVCHAR parameter and rerun the SRG job.

FABO9000A (AREADEF-008) ** INVALID OPTMZ PARAMETER

Explanation: The OPTMZ parameter specification on the AREADEF control statement is not correct.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the OPTMZ parameter specification, and rerun the SRG job. For specification of this parameter, see OPTMZ in "For SR and equation table generation method specifications" on page 40.

FABO9000A (AREADEF-009) ** [KEYRNG | ARETRY] PARAMETER UNMATCH WITH AREXT

Explanation: The AREXT parameter value and the number of pairs that are provided with the KEYRNG or ARETRY parameter are inconsistent. Consider the following example:

```
KEYRNG = (0000,2222),(0001,9999),(3333,4444), AREXT = 2
```

In this example, AREXT=2 indicates that the two pairs of figures in the parentheses will be used to specify one key range. Therefore, the system creates the first key range for the area as (00000001,22229999) by combining the first and second pairs, and then tries to create the second key range by using the third and fourth pairs. However, because a fourth pair is not specified, an error occurs.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Change the number of paired figures with the KEYRNG or ARETRY parameter, or change the value of the AREXT parameter so that they are consistent. In the preceding example, add one more pair of figures to the KEYRNG or ARETRY parameter, or change the AREXT parameter value to 1 or 3. Then, rerun the SRG job. For specification of these parameters, see KEYRNG or ARETRY and AREXT.

FABO9000A (AREADEF-010) ** LINKSR CONTROL STATEMENT EXISTENCE

Explanation: A LINKSR control statement preceded the AREADEF control statement. The control statements that may precede the AREADEF control statement are AREADEF and ANALYZE.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the case:

- When SR is to be generated in one job, move the LINKSR control statement after the AREADEF control statement, and rerun the SRG job.
- When SR is not to be generated in one job, remove the LINKSR control statement and rerun the SRG job.

FABO9000A (AREADEF-011) ** INVALID [KEYRNG | ARETRY] PARAMETER

Explanation: An error was detected in the KEYRNG or ARETRY parameter specification of the AREADEF control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify a correct parameter, and rerun the SRG job.

FABO9000A (AREADEF-012) ** INVALID TLIMIT PARAMETER

Explanation: An error was detected in the TLIMIT parameter specification of the AREADEF control statement, or, the TLIMIT parameter was specified without the OPTMZ parameter specification.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify a correct parameter, and rerun the SRG job.

FABO9000A (AREADEF-013) ** IGNORE BECAUSE SRCMIG=YES

Explanation: SRCMIG=YES was specified on the ANALYZE control statement, and the AREADEF control statement was specified at the same time. The AREADEF control statement cannot be specified when SRCMIG=YES.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify SRCMIG=NO or delete the AREADEF control statement, and rerun the SRG job.

FABO9000A (AREADEF-014) ** MORE THAN 3000 COMBINATIONS OF (A,B) IN [KEYRNG | ARETRY]

Explanation: More than 3000 combinations of (A,B) were specified in the KEYRNG or ARETRY parameter.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Reduce the number of range of keys and run SRG again.

FABO9000A (AREADEF-015) ** IGNORE BECAUSE TYPECHG=YES

Explanation: TYPECHG=YES was specified in the ANALYZE control statement, and the AREADEF control statement was specified at the same time. The AREADEF control statement cannot be specified when TYPECHG=YES.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify TYPECHG=NO or delete the AREADEF control statement, and rerun the SRG job.

FABO9000A (AREADEF-016) ** INVALID TNUMLIM PARAMETER

Explanation: One of the following conditions exists:

- The TNUMLIM parameter on the AREADEF control statement has an error.
- Both the TNUMLIM parameter and the TLIMIT parameter were specified.
- The TNUMLIM parameter was specified without an OPTMZ parameter.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the parameters and rerun the SRG job.

FABO9000A (AREADEF-017) ** INVALID OPTMTIME PARAMETER

Explanation: Either an error was detected in the OPTMTIME parameter of the AREADEF control statement, or the OPTMTIME parameter was specified without an OPTMZ parameter.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the parameters and rerun the SRG job.

FABO9000A (AREADEF-018) ** KEYRNG PARAMETER CANNOT BE SPECIFIED WITH ARETRY PARAMETER

Explanation: The KEYRNG parameter and the ARETRY parameter of the AREADEF control statement are mutually exclusive.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Specify one of these parameters and rerun SRG.

FABO9000A (LINKSR-001) ** LINKSR CONTROL STATEMENT IS DUPLICATED

Explanation: More than one LINKSR control statement exists. The system permits only one LINKSR control statement for each SRG execution.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Delete the unnecessary LINKSR control statement(s) and rerun the SRG job.

FABO9000A (LINKSR-002) ** ANALYZE CONTROL STATEMENT IS NOT FOUND

Explanation: The ANALYZE control statement does not precede the LINKSR control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- When the ANALYZE control statement exists after the LINKSR control statement, move the ANALYZE control statement before the LINKSR control statement and rerun the SRG job.
- When no ANALYZE control statement exists, code one before the LINKSR control statement and rerun the SRG job.

FABO9000A (LINKSR-003) ** IGNORED BECAUSE TYPE IS *xxxx*

Explanation: The LINKSR control statement was specified, but a database type other than DEDB was specified as the TYPE parameter of the ANALYZE control statement. The LINKSR control statement can be specified only when TYPE=DEDB. *xxxx* is the database type specified with the TYPE parameter of the ANALYZE control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- If the LINKSR control statement was coded for the database type VSAM or OSAM, delete the LINKSR control statement and rerun the SRG job.
- If a wrong database type was specified on the ANALYZE control statement, correct the database type to TYPE=DEDB, and rerun the SRG job.

FABO9000A (LINKSR-004) ** AREAID PARAMETER IS NOT FOUND

Explanation: The AREAID parameter was not specified on the LINKSR control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the AREAID parameter and rerun the SRG job. For specification of the AREAID parameter, see AREAID.

FABO9000A (LINKSR-005) ** NUMBER OF AREA IS OVER 240

Explanation: The number of area names specified with the AREAID parameter exceeds 240. SRG limits the number of areas for one database to 240 to match the DEDB specifications.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- If you tried to create a database with more than 240 areas by the design error, decrease the number of areas to 240 or less by combining some of them. For this, you must provide a key file for the areas to be combined, decrease the number of area names specified with the AREAID parameter, and rerun the SRG job.
- If unnecessary area names are specified with the AREAID parameter, remove the unnecessary area names and rerun the SRG job.

FABO9000A (LINKSR-006) ** NUMBER OF AREA UNMATCH WITH NOAREA OF ANALYZE

Explanation: The number of area names specified with the AREAID parameter did not match the value specified with the NOAREA parameter of the ANALYZE control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Make both numbers match and rerun the SRG job.

FABO9000A (LINKSR-007) ** AREA *name* ON LINKID IS NOT FOUND IN AREAID PARAMETER

Explanation: The DEDB area name (*name*) specified in the LINKID parameter was not specified in the AREAID parameter.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the LINKID parameter and rerun the SRG job.

FABO9000A (LINKSR-008) ** LINKID PARAMETER IS NOT ALLOWED FOR SRCMIG=YES

Explanation: SRCMIG=YES was specified on the ANALYZE control statement, and the LINKID parameter was specified on the LINKSR control statement. The LINKID parameter cannot be specified when SRCMIG=YES.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the LINKID parameter or SRCMIG parameter, and rerun the SRG job.

FABO9000A (LINKSR-009) ** LINKID PARAMETER IS NOT ALLOWED FOR TYPECHG=YES

Explanation: TYPECHG=YES was specified in the ANALYZE control statement, and the LINKID parameter was specified on the LINKSR control statement. The LINKID parameter cannot be specified when TYPECHG=YES.

System action: SRG issues the FABO0107A message and ends the job with RC=8.

User response: Correct the LINKID parameter or TYPECHG parameter, and rerun the SRG job.

FABO9000A (LINKSR-010) ** INVALID VALUE IS SPECIFIED FOR SRCCHK PARAMETER

Explanation: An incorrect value is specified for the SRCCHK parameter of the LINKSR control statement.

System action: SRG issues message FABO0107A and ends with RC=8.

User response: Correct the SRCCHK parameter, and rerun the SRG job.

FABO9000A (LINKSR-011) ** NONZERO SRCCHK PARAMETER IS NOT ALLOWED FOR SRCMIG=YES

Explanation: You are migrating the SR sources, but an SRCCHK option greater than 0 is specified. SRCCHK parameter values other than 0 are valid only for migrated SR sources.

System action: SRG issues message FABO0107A and ends with RC=8.

User response: If you want to migrate your old SR sources to the new format, remove the SRCCHK parameter from the LINKSR control statement or specify SRCCHK=0, and rerun the SRG job. If your SR sources have already been migrated, remove the SRCMIG=YES parameter from the ANALYZE control statement, and rerun the SRG job.

FABO9000A (XNUMCK-001) ** *xxxxxxxx* PARAMETER IS NOT FOUND

Explanation: The *xxxxxxxx* parameter was not specified on the control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the *xxxxxxxx* parameter on the control statement and rerun the SRG job.

FABO9000A (XNUMCK-002) ** LOW PARAMETER IS NOT NUMERIC

Explanation: An unexpected error occurred.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Contact your IBM representative with the SRG control statement assembly listing.

FABO9000A (XNUMCK-003) ** HIGH PARAMETER IS NOT NUMERIC

Explanation: An unexpected error occurred.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Contact your IBM representative with the SRG control statement assembly listing.

FABO9000A (XNUMCK-004) ** RANGE PARAMETER IS NOT NUMERIC

Explanation: An unexpected error occurred.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Contact your IBM representative with the SRG control statement assembly listing.

FABO9000A (XNUMCK-005) ** *xxxxxxxx* **PARAMETER IS LOWER THAN** *yyyyyyyy*

Explanation: The value specified with the *xxxxxxxx* parameter is smaller than *yyyyyyyy*. *xxxxxxxx* represents the parameter name, and *yyyyyyyy* indicates the minimum value that can be specified.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Change the value of the *xxxxxxxx* parameter to be greater than or equal to *yyyyyyyy* and rerun the SRG job.

FABO9000A (XNUMCK-006) ** *xxxxxxxx* **PARAMETER IS HIGHER THAN** *yyyyyyyy*

Explanation: The value specified with the *xxxxxxxx* parameter is greater than *yyyyyyyy*. *xxxxxxxx* represents the parameter name, and *yyyyyyyy* indicates the maximum value that can be specified.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Change the value of the *xxxxxxxx* parameter to be smaller than or equal to *yyyyyyyy* and rerun the SRG job.

FABO9000A (XNUMCK-007) ** *xxxxxxxx* **PARAMETER IS OUT OF RANGE**

Explanation: The numeric value specified with the *xxxxxxxx* parameter is outside the specifiable range.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Change the value of the *xxxxxxxx* parameter so that it is within the valid range, and rerun the SRG job. For specification of the range, see “Control statements” on page 28.

FABO9000A (XNUMCK-008) ** *xxxxxxxx* **PARAMETER IS NOT NUMERIC**

Explanation: The value specified with the *xxxxxxxx* parameter is not numeric.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the *xxxxxxxx* parameter to a numeric value and rerun the SRG job.

FABO9000A (XNUMCK-009) ** *xxxxxxxx* **IS INVALID PARAMETER WHEN TYPE IS** *yyyyyyyy*

Explanation: The specified parameter *xxxxxxxx* is incorrect for the database type *yyyyyyyy*.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Take one of the following actions depending on the error type:

- If the specification of the database type was wrong, correct the TYPE parameter of the ANALYZE control statement, and rerun the SRG job.
- If the *xxxxxxxx* parameter was specified by mistake, remove the parameter and rerun the SRG job.

FABO9000A (XCOMP-001) ** COMPRESSION PARAMETER NOT FOUND

Explanation: A key attribute was not specified with the key compression parameter of the ANALYZE control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the key compression parameter of the ANALYZE control statement and rerun the SRG job. For specification of the key compression parameter, see “For input database key file specifications” on page 33.

FABO9000A (XCOMP-002) ** COMPRESSION PARAMETER IS INVALID IN *xxx*

Explanation: An error was found in the specification of the *xxx*th key compression parameter of the ANALYZE control statement. *xxx* shows the sequence number, counted from the beginning, of the key compression parameter in which the error was detected.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the incorrect key compression parameter and rerun the SRG job.

FABO9000A (XCOMP-003) ** COMPRESSION PARAMETER IS NOT NUMERIC

Explanation: A nonnumeric offset was specified with the key compression parameter of the ANALYZE control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the incorrect key compression parameter and rerun the SRG job. For specification of the key compression parameter, see “For input database key file specifications” on page 33.

FABO9000A (XCOMP-004) ** PACK FIELD WITHOUT SIGN MUST BE WITHIN 4 BYTE LENGTH IN *xxx*

Explanation: Although the key attribute 'P' was specified for the *xxx*th key compression parameter of the ANALYZE control statement, its subkey field is longer than 4 bytes. When the key attribute is 'P,' the maximum length of one subkey field is 4 bytes.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Divide the incorrect subkey field into subkey fields of less than or equal to 4 bytes. An example is shown below:

(1, 8, P) → (1, 4, P), (5, 8, P)

In the above example, the subkey field of 8-byte unsigned packed decimal data is divided into two 4-byte subkey fields, and causes no error.

FABO9000A (XCOMP-005) ** PACK FIELD WITH SIGN MUST BE WITHIN 5 BYTE LENGTH IN *xxx*

Explanation: Although the key attribute 'S' was specified for the *xxx*th key compression parameter of the ANALYZE control statement, its subkey field is longer than 5 bytes. When the key attribute is 'S,' the maximum length of one subkey field is 5 bytes.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Divide the incorrect subkey field into subkey fields of less than or equal to 5 bytes. An example is shown below.

(1, 8, S) → (1, 3, P), (4, 8, S)

In the above example, the subkey field of 8-byte signed packed decimal data is divided into subkey field of 3-byte unsigned packed decimal data and subkey field of 5-byte signed packed decimal data. This will cause no error.

FABO9000A (XCOMP-006) ** NUMERIC FIELD MUST BE WITHIN 9 BYTE LENGTH IN *xxx*

Explanation: Although the key attribute 'N' was specified for the *xxx*th key compression parameter of the ANALYZE control statement, its subkey field is longer than 9 bytes. When the key attribute is 'N,' the maximum length of one subkey field is 9 bytes.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Divide the incorrect subkey field into subkey fields of less than or equal to 9 bytes. An example is shown below.

(1, 12, N) → (1, 9, N), (10, 12, N)

In the above example, the subkey field of 12-byte numeric data is divided into a 9-byte subkey field and 3-byte subkey field. This will cause no error.

FABO9000A (XCOMP-008) ** COMPRESSION PARAMETER TYPE IS INVALID IN *n*

Explanation: The key attribute specified in the *n*th key compression parameter ((a,b,c) or (a,b,c,d)) of the ANALYZE control statement is not correct.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the key compression parameter and rerun the SRG job.

FABO9000A (XCOMP-009) ** COMPRESSION PARAMETER LENGTH IS UNMATCH WITH LENGTH PARAMETER

Explanation: The sum of each subkey field length specified with the key compression parameter and the input key length specified with the INKEYL parameter of the ANALYZE control statement are not the same. An example is shown below:

(1, 4, P), (5, 8, N), INKEYL=7

In the above example, the sum of the subkey field lengths is (4 - 1 + 1) + (8 - 5 + 1) = 8. But the input key length is 7. Thus both do not match.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the key compression parameter or the INKEYL parameter and rerun the SRG job.

FABO9000A (XCOMP-010) ** COMPRESSION PARAMETER SPECIFICATION IS OVERLAP

Explanation: Some of the subkey fields specified with the compression parameter of the ANALYZE control statement overlap.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the specifications with the key compression parameter and rerun the SRG job.

FABO9000A (XCEXT-001) ** CONVERSION CHARACTER IS NOT HEX IMAGE

Explanation: The specific characters specified with the subkey field attribute 'U'on the ANALYZE control statement is not in correct hexadecimal format. An example is shown below:

(1, 4, U, F0F1FQ)

The fourth parameter 'F0F1FQ' indicates the specific characters, but it contains a nonhex character 'Q.' (In hex expression, all the characters are represented by a combination of 0 through 9, and A through F.)

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the specific characters and rerun the SRG job.

FABO9000A (XCEXT-002) ** CONVERSION VALUE IS DUPLICATE

Explanation: Duplicate expressions were found in the 'specific character' field specified with the subkey field attribute 'U' on the ANALYZE control statement. An example is shown below:

(1, 4, U, F0F1F1F2)

In this example, 'F1' is a duplicate.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the duplication and rerun the SRG job.

FABO9000A (XCEXT-003) ** CONVERSION CHARACTER MUST BE OVER 2 CHARACTERS

Explanation: Only one hex character was provided in the 'specific character' field specified with the subkey field attribute 'U' on the ANALYZE control statement. When 'U' is specified, more than one hex character must be provided. An example is shown below:

(1, 4, U, F0)

In this example, only 'F0' is provided for the specific character, thus causing an error.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify more than one hex character for the specific character and rerun the SRG job.

FABO9000A (XCEXT-004) ** CONVERSION CHARACTER LENGTH IS INVALID

Explanation: An error was found in the specific character field specified with the subkey field attribute 'U' on the ANALYZE control statement. An example is shown below:

(1, 4, U, F0F1F)

In the above example, the last F does not express a hexadecimal character. (A hexadecimal character is always represented with two characters; therefore, the number of characters specified must always be even.)

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the specification and then rerun the SRG job.

FABO9000A (XALPHACK-001) ** ***xxxxxxxx*** **PARAMETER IS NOT FOUND**

Explanation: The *xxxxxxxx* parameter was not specified on the control statement. *xxxxxxxx* is the parameter name that must be specified.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the *xxxxxxxx* parameter on the control statement and rerun the SRG job.

FABO9000A (XALPHACK-002) ** DUPCK PARAMETER IS INVALID

Explanation: An unexpected error occurred.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Contact your IBM representative with the SRG control statement assembly listing.

FABO9000A (XALPHACK-003) ** ***xxxxxxxx*** **PARAMETER IS CONSTRUCTED OF INVALID CHARACTER**

Explanation: An incorrect character was used to specify the *xxxxxxxx* parameter. *xxxxxxxx* represents the parameter name. The characters that can be used for this parameter are shown below:

1st character:
A - Z, @, # (Total 29 characters)

2nd - 8th characters:
A - Z, @, #, 0 - 9 (Total 39 characters)

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the incorrect character and then rerun the SRG job.

FABO9000A (XALPHACK-004) ** *xxxxxxxx* PARAMETER IS INVALID

Explanation: A character not permissible with SRG was found in the *xxxxxxxx* parameter. *xxxxxxxx* indicates the name of the error-detected parameter.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the parameter specification parameter and rerun the SRG job.

FABO9000A (XALPHACK-005) ** DUPLICATED PARAMETER VALUE: *xxxxxxxx*

Explanation: The parameter value *xxxxxxxx* was specified in another parameter. An example is shown below:

```
AREA1   AREADEF
AREA1   AREADEF
```

In the above example, the label AREA1 is a duplicate. In this case, the following message is issued:

```
XALPHACK-005 ** PARAMETER VALUE DUPLICATE
NAME= AREA1
```

This message is issued also when a name of an area is specified with the name of the SR. This message, in this case, means that the SR name specification is superfluous. The user need not to specify explicitly the SR name in LINKID parameter, since, if at least one area name is specified on LINKID parameter, then the logic part of the SR is re-assembled and re-linkedited automatically.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the duplication and rerun the SRG job.

FABO9000A (XALPHACK-006) ** *xxxxxxxx* PARAMETER LENGTH IS OVER 8

Explanation: More than eight characters were specified with the *xxxxxxxx* parameter. *xxxxxxxx* indicates the name of the error-detected parameter.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Correct the specification of the *xxxxxxxx* parameter to eight or less characters and rerun the SRG job.

FABO9000A (XARCHK-001) ** [KEYRNG | ARETRY] PARAMETER IS NOT MATCH WITH COMPRESSION PARAMETER

Explanation: The KEYRNG parameter or the ARETRY parameter value does not match the key compression parameter of the ANALYZE control statement.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Specify the KEYRNG parameter or the ARETRY parameter correctly and rerun the SRG job. For more information, see KEYRNG or ARETRY.

FABO9000A (XPRI-001) ** HIGHKEY IS LOWER THAN LOWKEY

Explanation: The key range specification in the KEYRNG parameter or the ARETRY parameter of the AREADEF control statement is not correct. The minimum value of a key range is larger than the maximum value in the range.

System action: SRG issues the FABO0109A message and ends with RC=8.

User response: Correct the KEYRNG parameter or the ARETRY parameter value of the AREADEF control statement for the area that has the error and rerun the SRG job for the area.

FABO9000A (XPRI-002) ** LOWKEY IS LOWER THAN PREVIOUS HIGHKEY

Explanation: The key range specification in the KEYRNG parameter or the ARETRY parameter of the AREADEF control statement is not correct. The minimum value of a key range is smaller than the maximum value of the previous key range.

System action: SRG issues the FABO0109A message and ends with RC=8.

User response: Correct the KEYRNG parameter or the ARETRY parameter value of the AREADEF control statement for the area that has the error and rerun the SRG job for the area.

FABO9000A (XNOTE-001) ** TEXT FORMAT ERROR

Explanation: An unexpected error occurred.

System action: SRG issues the FABO0107A message and ends with RC=8.

User response: Contact your IBM representative with the SRG control statement assembly listing.

How to look up message explanations

You can use several methods to search for messages and codes.

Searching for messages on the Web

You can use any of the popular search engines that are available on the Web to search for message explanations. When you type the specific message number or code into the search engine, you will be presented with links to the message information in IBM Knowledge Center.

Gathering diagnostic information

Before you report a problem with IMS Sequential Randomizer Generator to IBM Software Support, you need to gather the appropriate diagnostic information.

Procedure

Provide the following information for all IMS Sequential Randomizer Generator 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.
- A Load Module/Macro APAR Status report.

For information about creating a Load Module/Macro APAR Status report, see "Diagnostics aid" on page 169.

Diagnostics aid

If you have a problem that you think is not a user error, you should run the Diagnostics Aid (FABODIAG), obtain the Load Module/Macro APAR Status report, attach it to the other diagnostic documents (such as job dump list or I/O of the utility), and report the error to IBM.

The Diagnostics Aid generates a Load Module/Macro APAR Status report. This report shows the latest APAR fixes applied to each module and macro.

The Diagnostics Aid is not applicable for any other versions or releases.

How to run diagnostics aid with JCL

To run the Diagnostics Aid (FABODIAG), supply an EXEC statement and a DD statement that defines the output data set.

EXEC
This statement must be in the following form:

```
//stepname  EXEC PGM=FABODIAG
```

STEPLIB DD
This statement points to the load module library data set where FABODIAG load module resides:

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

where HPS.SHPSLMD0 is the name of the library that contains the IMS Sequential Randomizer Generator load modules.

SHPSLMD DD
This statement defines the library containing the load modules (usually HPS.SHPSLMD0) for which you have a problem.

The Load Module APAR Status report is not generated if this DD statement is not provided or if DD DUMMY is specified.

It is always recommended that you specify this DD statement.

SHPSMAC DD
This statement defines the library containing the provided macros (usually HPS.SHPSMAC0) for which you have a problem.

The Macro APAR Status report is not generated if this DD statement is not provided or if DD DUMMY is specified.

SYSPRINT DD
This output data set contains the Load Module/Macro APAR Status report. The data set contains 133-byte, fixed-length records. It can reside on a tape, direct-access device, or printer; or it can be routed through the output stream. If BLKSIZE is coded in the DD statement, it must be a multiple of 133. However, it is recommended that you use:

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

Load Module/Macro APAR Status report

The Diagnostics Aid generates two reports for maintenance by IBM.

The generated reports are:
- Load Module APAR Status report
- Macro APAR Status report

Load Module APAR Status report

The Load Module APAR Status report contains information about the modules and their applied APARs.

This report contains the following information:

MODULE LIBRARY
This includes the data set names specified in the SHPSLMD DD statement. If more than 30 data sets are concatenated, only the first 30 data sets are listed.

MODULE NAME
This is the name of the load module member or the alias.

ALIAS-OF
This is the name of the original member of the alias. If the module name is not an alias, this field is left blank.

CSECT NAME
This is the name of the included CSECT in the module. The CSECT names are reported in the included order in the module.

APAR NUMBER
This is the latest APAR number applied to the module represented by the CSECT name. If no APAR is applied, NONE is shown.

APAR FIX-DATE
This is the date when the modification was prepared for the module represented by the CSECT name. If no APAR is applied, N/A is shown.

Note:
1. If the CSECT name does not start with *FAB, HPS,* or the program structure of the CSECT does not conform to the IMS Sequential Randomizer Generator module standard to identify the APAR number and the APAR fixed date, the fields APAR NUMBER and APAR FIX-DATE are filled with asterisks (*).
2. If the load module is a member of the PDSE library, the following statement is shown on the report line and the job completes with a return code of 4.
```
** IT CAN NOT BE ANALYZED DUE TO PDSE LIBRARY MEMBER **
```
3. If the load macro fails for a utility member, the following statement is shown on the report line and the job completes with a return code of 8.
```
** IT CAN NOT BE ANALYZED DUE TO LOAD FAILED MEMBER  **
```

Macro APAR Status report

The Macro APAR Status report contains information about macros and their applied APARs.

This report contains the following information:

MACRO LIBRARY
This includes the data set names specified in the SHPSMAC DD statement. If more than 30 data sets are concatenated, only the first 30 data sets are listed.

MACRO NAME
This is the name of the macro member or the alias.

ALIAS-OF
This is the name of the original member of the alias. If the macro name is not an alias, this field is left blank.

APAR NUMBER
This is the latest APAR number applied to the macro. If no APAR is applied, NONE is shown.

APAR FIX-DATE
This is the date when the modification was prepared for the macro. If no APAR is applied, N/A is shown.

Note: If the macro source statement structure does not conform to the IMS Sequential Randomizer Generator macro standard to identify the APAR number and the APAR fixed date, the fields APAR NUMBER and APAR FIX-DATE are filled with asterisks (*).

Messages and codes

The following topics describe the return codes, abend codes, and messages issued by the Diagnostics Aid.

Return codes

FABODIAG contains the following return codes:

0 Successful completion of the program.

4 Warning messages were issued, but the requested operation was completed.

8 Error messages were issued, but the request operation was completed.

Abend codes

All 36*xx* abend codes are accompanied by an FABU36*xx* message. Refer to the appropriate message for problem determination.

Messages

Use the information in these messages to help you diagnose and solve FABODIAG problems.

FABU1001I DIAG ENDED NORMALLY

Explanation: This message is generated when Diagnostic Aid has been completed successfully.

System action: Diagnostic Aid completes the job successfully with a return code of 0.

User response: None. This message is informational.

FABU1002W DIAG ENDED WITH WARNINGS

Explanation: This message is generated when trivial error conditions are encountered by Diagnostic Aid.

System action: Diagnostic Aid ends with a return code of 4.

User response: Refer to other messages generated by Diagnostic Aid to determine the nature and the cause of the detected errors. Correct the problem and rerun the job.

FABU1003E DIAG ENDED WITH ERRORS

Explanation: This message is generated when severe error conditions are encountered by Diagnostic Aid.

System action: Diagnostic Aid ends with a return code of 8.

User response: Refer to other messages generated by Diagnostic Aid to determine the nature and the cause of the detected errors. Correct the problem and rerun the job.

FABU1005W [SHPSLMD | SHPSMAC] DD STATEMENT NOT FOUND

Explanation: Diagnostic Aid could not find the SHPSLMD/SHPSMAC DD statement.

System action: Diagnostic Aid sets an end-of-job return code of 4 and continues processing. Diagnostic Aid does not generate a report for the load module or the macro.

User response: If you intended to specify the indicated DD statement, correct the error and rerun the job.

FABU1006W DUPLICATE *member name* **IN LIBRARY DDNAME** *ddname*

Explanation: Diagnostic Aid found a duplicated member in the concatenated libraries.

System action: Diagnostic Aid uses the member which is first found in the concatenated libraries. Diagnostic Aid sets an end-of-job return code of 4 and continues processing.

User response: Ensure which libraries have correct module/macro libraries. Correct the error and rerun the job if necessary.

FABU1007W DUMMY SPECIFIED FOR [SHPSLMD | SHPSMAC] DD STATEMENT

Explanation: DUMMY was specified for the SHPSLMD/SHPSMAC DD statement.

System action: Diagnostic Aid sets an end-of-job return code of 4 and continues processing. Diagnostic Aid does not generate a report for the load module or the macro.

User response: If you did not intend to specify the dummy DD statement, correct the error and rerun the job.

FABU1008W NO [MODULE | MACRO] MEMBERS FOUND IN DDNAME [SHPSLMD | SHPSMAC]

Explanation: Diagnostic Aid could not find any utility modules or macros members from the DD ddname data set.

System action: Diagnostic Aid sets an end-of-job return code of 4 and continues processing.

User response: Ensure that the libraries have correct utility module or macro libraries. Correct the error and rerun the job.

FABU2001E LOAD FAILED FOR DDNAME *ddname* **MODULE** *member*

Explanation: Diagnostic Aid could not load a *member name* from *ddname*.

System action: Diagnostic Aid sets an end-of-job return code of 8 and continues processing.

User response: Ensure that the member indicated exists in the data set specified for the indicated *ddname*. Correct the error and rerun the job.

FABU3600E OPEN FAILED FOR DDNAME *ddname*

Explanation: The named DCB could not be opened.

System action: Diagnostic Aid ends with an abend code of U3600.

User response: Ensure that a *ddname* DD statement exists, and that it specifies the correct DD parameter. Correct any errors, and rerun the job.

FABU3601E GET FAILED FOR DDNAME *ddname*

Explanation: The GET failed for a directory from the DD *ddname* data set.

System action: Diagnostic Aid ends with an abend code of U3601.

User response: Refer to the MVS system message and its programmer response. Correct the error and rerun Diagnostic Aid. If the error persists, contact IBM Software Support.

FABU3602E READ FAILED FOR DDNAME *ddname* **MEMBER** *member*

Explanation: The READ failed for a *member* from the DD *ddname* data set.

System action: Diagnostic Aid ends with an abend code of U3602.

User response: Refer to the MVS system message and its programmer response. Correct the error and rerun Diagnostic Aid. If the error persists, contact IBM Software Support.

FABU3603E BLDL FAILED FOR DDNAME *ddname* **MEMBER** *member*

Explanation: The *member* was not found when the BLDL macro searched the PDS directory for the *ddname*.

System action: Diagnostic Aid ends with an abend code of U3603.

User response: Ensure that the member indicated exists in the data set specified for the indicated ddname. Correct the error and rerun the job. If the error persists, contact IBM Software Support.

FABU3604E LOAD FAILED FOR DDNAME *ddname* **MODULE** *member*

Explanation: Diagnostic Aid could not load the *member name* from the *ddname*.

System action: Diagnostic Aid ends with an abend code of U3604.

User response: Refer to the MVS system message and its programmer response. Correct the error and rerun Diagnostic Aid. If the error persists, contact IBM Software Support.

FABU3605E DELETE FAILED FOR MODULE *member*

Explanation: Diagnostic Aid could not delete a *member name*.

System action: Diagnostic Aid ends with an abend code of U3605.

User response: Contact IBM Software Support.

FABU3606E PUT FAILED FOR SYSPRINT

Explanation: Diagnostic Aid could not put report data in SYSPRINT.

System action: Diagnostic Aid ends with an abend code of U3606.

User response: Refer to the MVS system message and its programmer response. Correct the error and rerun Diagnostic Aid. If the error persists, contact IBM Software Support.

FABU3607E OPEN FAILED FOR SYSPRINT

Explanation: SYSPRINT DCB could not be opened.

System action: Diagnostic Aid ends with an abend code of U3607.

User response: Ensure that a *ddname* SYSPRINT DD statement exists, and that it specifies the correct DD parameter. Correct any errors, and rerun the job.

FABU3608E FIND FAILED FOR DDNAME *ddname* **MEMBER** *member*

Explanation: The FIND failed for a *member* from DDNAME *ddname* data set.

System action: Diagnostic Aid ends with an abend code of U3608.

User response: Ensure that the member indicated exists in the data set specified for the indicated ddname. Correct the error and rerun the job. If the error persists, contact IBM Software Support.

FABU3609E DEVTYPE FAILED FOR DDNAME *ddname*

Explanation: The DEVTYPE failed for a DDNAME *ddname* data set.

System action: Diagnostic Aid ends with an abend code of U3609.

User response: Contact IBM Software Support.

FABU3610E RDJFCB FAILED FOR DDNAME *ddname*

Explanation: The READJFCB failed for a DDNAME *ddname* data set.

System action: Diagnostic Aid ends with an abend code of U3610.

User response: Contact IBM Software Support.

FABU3611E GETMAIN FAILED. INSUFFICIENT STORAGE TO RUN THE JOB

Explanation: Work space for Diagnostic Aid could not be obtained.

System action: Diagnostic Aid ends with an abend code of U3611.

User response: Increase the region size and rerun the job.

FABU3612E TOO MANY [MODULE | MACRO] MEMBERS DETECTED IN DDNAME [SFABMOD | SHPSMAC]

Explanation: There are too many utility members in the SFABMOD/SHPSMAC DD data set.

System action: Diagnostic Aid ends with an abend code of U3612.

User response: Specify the correct data set for the indicated DD statement and rerun the job.

Notices

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the Unites States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at http://www.ibm.com/privacy and 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

Product Number: 5655-E11

Printed in USA

SC27-0926-03