

# Using GENJCL.USER to Allocate IMS HALDB Data Sets

## Overview

IMS V7 HALDB databases often contain many data sets. Allocating these data sets could be a time consuming process for two reasons. First, there may be many data sets. Second, the allocation jobs must use system-assigned data set names. IMS assigns these names during the partition definition process. This article explains how you may use DBRC GENJCL.USER commands to create data set allocation jobs easily.

There are four steps in the process.

### 1. Creation of Skeletal JCL

You create skeletal JCL members for the various types of database data sets. These types are indexes, Indirect List Data Sets (ILDSs), ESDS data sets, and OSAM data sets. The skeletal JCL members include the IDCAMS DEFINE statements for the VSAM data sets or ALLOCATE statements for OSAM data sets.

### 2. Definition of DBDS Groups

You define DBDS groups for each HALDB database when you define the database. There is one DBDS group for each of its data set types. For example, a PHIDAM database would have a group for its indexes, a group for its ILDSs, and a group for each of its data set groups.

### 3. Execution of GENJCL.USER Commands

You use GENJCL.USER commands to generate the allocation jobs. These commands specify the skeletal JCL members and DBDS groups defined in the previous steps. The commands also may specify user variables, such as primary and secondary allocation amounts.

### 4. Execution of Allocation Jobs

You execute the jobs generated by the previous step to allocate the data sets. If necessary, you may modify the jobs to change the allocations for specific partitions

A more complete explanation of these steps follows. The examples shown are meant to be illustrations of how you may implement these techniques. You should modify them for your environment.

## Creation of Skeletal JCL

A typical installation will want at least four skeletal JCL members. One member is used for allocating PHIDAM prime index and secondary index data sets. These are KSDSs. One member is used for allocating Indirect List Data Sets. These are KSDSs. One member is used for allocating VSAM "data" data sets. These are ESDSs. "Data" data sets are the data sets that contain segments in PHIDAM and PHDAM databases. One member is used for allocating OSAM "data" data sets.

The following shows how these skeletal JCL members could be defined. You should modify them to meet your installation's needs. For example, if SMS-managed storage is used, the values for STORCLAS and MGMTCLAS would have to meet your installation's standards. If you do not use SMS-managed storage, you would replace STORCLAS and MGMTCLAS with the appropriate VOLUMES parameter for your installation.

## Index Skeletal JCL Member

You may use the following member to create PHIDAM index and secondary index data sets. In the examples in this article, its member name is DBDDDEFX. It includes the following user variables:

%UPRIM	Primary space allocation amount in cylinders
%USEC	Secondary space allocation amount in cylinders
%UKSIZE	Size of key
%UKOFFST	Offset of key
%UCISZ	Data component CI size
%URECSZ	Record size
%UFREECI	CI free space percentage
%UFREECA	CA free space percentage

Member DBDDDEFX:

```
%DELETE (%STPNO NE '00000')
//S%STPNO EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
%ENDDEL
%SELECT DBDS((%DBNAME,%DDNAME))
  DEFINE CLUSTER (NAME(%DBDSN) -
    STORCLAS(STANDARD) -
    MGMTCLAS(STANDARD) -
    CYLINDERS (%UPRIM %USEC) -
    REUSE -
    SHR(3 3) -
    KEYS(%UKSIZE %UKOFFST) -
    SPEED -
    BWO(TYPEIMS) -
    RECORDSIZE(%URECSZ %URECSZ) -
    FREESPACE(%UFREECI %UFREECA) ) -
  INDEX(NAME(%DBDSN.INDEX)) -
  DATA(NAME(%DBDSN.DATA) -
    CISZ(%UCISZ) )
%ENDSEL
```

## ILDS Skeletal JCL Member

You may use the following member to create ILDS data sets. In the examples in this article, its member name is DBDDDEFL. It includes the following user variables:

%UPRIM	Primary space allocation amount in cylinders
%USEC	Secondary space allocation amount in cylinders
%UCISZ	Data component CI size

Member DBDDDEFL:

```

%DELETE (%STPNO NE '00000')
//S%STPNO EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
%ENDDEL
%SELECT DBDS((%DBNAME,%DDNAME))
  DEFINE CLUSTER (NAME(%DBDSN) -
    STORCLAS(STANDARD) -
    MGMTCLAS(STANDARD) -
    CYLINDERS (%UPRIM %USEC) -
    RECORDSIZE(50 50) -
    FREESPACE(25 10) -
    SHR(3 3) -
    SPEED -
    KEYS(9 0)) -
  INDEX(NAME(%DBDSN.INDEX)) -
  DATA(NAME(%DBDSN.DATA) -
    CISZ(%UCISZ) )
%ENDSEL

```

## VSAM Data Skeletal JCL Member

You may use the following member to create VSAM “data” data sets. In the examples in this article, its member name is DBDDDEFV. It includes the following user variables:

```

%UPRIM      Primary space allocation amount in cylinders
%USEC       Secondary space allocation amount in cylinders
%UCISZ      CI size
%URECSZ     Record size

```

Member DBDDDEFV:

```

%DELETE (%STPNO NE '00000')
//S%STPNO EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
%ENDDEL
%SELECT DBDS((%DBNAME,%DDNAME))
  DEFINE CLUSTER (NAME(%DBDSN) -
    STORCLAS(STANDARD) -
    MGMTCLAS(STANDARD) -
    CYLINDERS(%UPRIM %USEC) -
    REUSE -
    SHR(3 3) -
    CISZ(%UCISZ) -
    RECORDSIZE(%URECSZ %URECSZ) -
    NONINDEXED SPEED ) -
  DATA(NAME(%DBDSN.DATA))
%ENDSEL

```

## OSAM Data Skeletal JCL Member

You may use the following member to create OSAM “data” data sets. In the examples in this article, its member name is DBDDDEFO. It includes the following user variables:

```

%UPRIM      Primary space allocation amount in cylinders

```

%USEC            Secondary space allocation amount in cylinders

Member DBDDDEFO:

```
%DELETE (%STPNO NE '00000')
//S%STPNO EXEC PGM=IDCAMS,DYNAMNBR=100
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
%ENDDEL
%SELECT DBDS((%DBNAME,%DDNAME))
  ALLOCATE -
    DSNAME('%DBDSN') -
    FILE(%DDNAME) -
    STORCLAS(STANDARD) -
    MGMTCLAS(STANDARD) -
    DSORG(PS) -
    NEW CATALOG -
    SPACE (%UPRIM %USEC) CYLINDERS
%ENDSEL
```

Defaults Member

You may use the following member to contain default values for user variables. In the examples in this article, its member name is DBDDFLTD. The GENJCL.USER commands specify this member in their DEFAULTS parameter.

Member DBDDFLTD

```
%UCISZ = '4096'
%UFREECI = '40'
%UFREECA = '10'
```

## Definition of DBDS Groups

The second step in the process is the definition of DBDS groups for each HALDB database. Typically, you would create the groups for a database when you define the database. These groups are used with GENJCL.USER commands to create the allocation jobs. There is one DBDS group for each of its data set types. A PHIDAM database would have a DBDS group for its indexes, a group for its ILDSs, and a group for each of its data set groups. That is, you would create a DBDS group for the A data sets, another for the B data sets, etc. A PHDAM database would have a DBDS group for its ILDSs, and a group for each of its data set groups. A PSINDEX database would have a DBDS group for its index data sets.

The following is an example of a PHIDAM database with two data set groups and a secondary index. The database name is RZLDBT. Its partition names are DBTP01, DBTP02, DBTP03, DBTP04, and DBTP05. The secondary index is RZLSIT1. Its partition names are SIT1P01, SIT1P02, SIT1P03, and SIT1P04. The HALDB naming convention creates DD names from these partition names. For example, the DD name of the ILDS data set for the DBTP01 partition is DBTP01L. The DD name for the PHIDAM index of this partition is DBTP01X. The DD names of the "data" data sets are DBTP01A and DBTP01B. The following DBRC commands create the four DBDS groups for this database and the DBDS group for the secondary index. The groups have been named to indicate the database and the type of data sets in the group. For example, DBDS group RZLDBTA contains the A data sets for the partitions in database RZLDBT.

```
//DBDSGRP JOB (999,XXX),
```

```

//          'INIT DBDSGRPS',
//          CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1),
//          NOTIFY=LEWIS,
//          REGION=4M
//D          EXEC PGM=DSPURX00
//STEPLIB DD DISP=SHR,DSN=IMS.SDFSRESL
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//IMS      DD DISP=SHR,DSN=IMS.DBDLIB
//*****
//*
//SYSIN DD *
INIT.DBDSGRP GRPNAME(RZLDBTA) MEMBERS( -
(DBTP01 ,DBTP01A), -
(DBTP02 ,DBTP02A), -
(DBTP03 ,DBTP03A), -
(DBTP04 ,DBTP04A), -
(DBTP05 ,DBTP05A))
INIT.DBDSGRP GRPNAME(RZLDBTB) MEMBERS( -
(DBTP01 ,DBTP01B), -
(DBTP02 ,DBTP02B), -
(DBTP03 ,DBTP03B), -
(DBTP04 ,DBTP04B), -
(DBTP05 ,DBTP05B))
INIT.DBDSGRP GRPNAME(RZLDBTL) MEMBERS( -
(DBTP01 ,DBTP01L), -
(DBTP02 ,DBTP02L), -
(DBTP03 ,DBTP03L), -
(DBTP04 ,DBTP04L), -
(DBTP05 ,DBTP05L))
INIT.DBDSGRP GRPNAME(RZLDBTX) MEMBERS( -
(DBTP01 ,DBTP01X), -
(DBTP02 ,DBTP02X), -
(DBTP03 ,DBTP03X), -
(DBTP04 ,DBTP04X), -
(DBTP05 ,DBTP05X))
INIT.DBDSGRP GRPNAME(RZLSIT1A) MEMBERS( -
(SIT1P01 ,SIT1P01A), -
(SIT1P02 ,SIT1P02A), -
(SIT1P03 ,SIT1P03A), -
(SIT1P04 ,SIT1P04A))

```

## Execution of GENJCL.USER Commands

The third step in the process is the use of GENJCL.USER commands to generate the allocation jobs. User variables, such as primary and secondary allocation amounts, must be specified on the GENJCL.USER commands.

Each GENJCL.USER command opens, writes, and closes the output data set. For this reason, each command uses a different member of data set IMS.JCLOUT. If multiple commands wrote to the same member, later commands would overwrite the output of earlier commands.

The following example shows GENJCL.USER commands that use the DBDS groups and skeletal JCL that were created in the first two steps. OSAM is used for the "data" data sets. Skeletal JCL member DBDDDEFO is used for these data sets. Skeletal JCL member DBDDDEFX is used for the PHIDAM index data sets and the secondary index data sets. The keys of the PHIDAM index are 5 bytes long at offset 5. The keys of the secondary index are 5 bytes long at offset 34. These values are specified for the %UKSIZE and %UKOFFST user variables. The PHIDAM index record size is 10 bytes. The secondary index record

size is 40 bytes. These values are specified for the %URECSZ user variable. Similarly, space and free space variables are also specified. Skeletal member DBDDDEFL is used for the ILDSs.

```
//GENJCLT JOB (999,XXX),
//          'GENJCL.USER',
//          CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1),
//          NOTIFY=LEWIS,
//          REGION=4M
//D          EXEC PGM=DSPURX00
//STEPLIB DD DISP=SHR,DSN=IMS.SDFSRESL
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//IMS DD DISP=SHR,DSN=IMS.DBDLIB
//JCLPDS DD DISP=SHR,DSN=IMS.JCLPDS
//JCLOUTA DD DISP=SHR,DSN=IMS.JCLOUT(ALLOCTA)
//JCLOUTB DD DISP=SHR,DSN=IMS.JCLOUT(ALLOCTB)
//JCLOUTX DD DISP=SHR,DSN=IMS.JCLOUT(ALLOCTX)
//JCLOUTL DD DISP=SHR,DSN=IMS.JCLOUT(ALLOCTL)
//JCLOUTS DD DISP=SHR,DSN=IMS.JCLOUT(ALLOCTS)
/*****
//*
//SYSIN DD *
GENJCL.USER GROUP(RZLDBTA) MEMBER(DBDDDEFO) JCLOUT(JCLOUTA) LIST -
USERKEYS((%UPRIM,'2'),(%USEC,'1')) -
DEFAULTS(DBDDFLTD) ONEJOB
GENJCL.USER GROUP(RZLDBTB) MEMBER(DBDDDEFO) JCLOUT(JCLOUTB) LIST -
USERKEYS((%UPRIM,'2'),(%USEC,'1')) -
DEFAULTS(DBDDFLTD) ONEJOB
GENJCL.USER GROUP(RZLDBTX) MEMBER(DBDDDEFX) JCLOUT(JCLOUTX) LIST -
USERKEYS((%UPRIM,'2'),(%USEC,'1'),(%UKSIZE,'5'), -
(%UKOFFST,'5'),(%UCISZ,'4096'),(%URECSZ,'10')) -
DEFAULTS(DBDDFLTD) ONEJOB
GENJCL.USER GROUP(RZLDBTL) MEMBER(DBDDDEFL) JCLOUT(JCLOUTL) LIST -
USERKEYS((%UPRIM,'1'),(%USEC,'1'))
DEFAULTS(DBDDFLTD) ONEJOB
GENJCL.USER GROUP(RZLSIT1A) MEMBER(DBDDDEFX) JCLOUT(JCLOUTS) LIST -
USERKEYS((%UPRIM,'1'),(%USEC,'1'),(%UKSIZE,'5'), -
(%UKOFFST,'34'),(%UCISZ,'4096'),(%URECSZ,'40')) -
DEFAULTS(DBDDFLTD) ONEJOB
```

## Execution of Allocation Jobs

The final step in the process is the execution of the allocation jobs generated in the third step. In the example, the JCL for the jobs is written to members ALLOCTA, ALLOCTB, ALLOCTX, ALLOCTL, and ALLOCTS in IMS.JCLOUT. The process produces the same allocation parameters for the data sets in every partition. For example, the A data sets for every partition will have the same space specifications. If one or more of your partitions require different parameters, you would have to modify the allocation specifications for such partitions before you execute the jobs.