Utilities for z/OS (OS/390) Disk Data Sets

This document describes utilities and JCL procedures for creating, modifying, copying, printing, and moving OS disk data sets, including CARDDISK, CARDLIST, DELETE, DFHEOFMT, DFHMDSET, DSNCHECK, DSAT, REPRO (IDCAMS), PRINT (IDCAMS), and TOCMS.

See also the z/OS (OS/390) Disk Data Sets at CNS [http://docweb.cns.ufl.edu/docs/d0045/] (D0045) manual for detailed information about using OS data sets and Utilities for Partitioned Data Sets [http://docweb.cns.ufl.edu/docs/d0068/] (D0068) for PDS utilities.

Related search terms include , storage, services, MVS, PDS, JCL, TSO, IDCAMS, and VSAM.

This material replaces and supplements information that was originally published in the CNS Utilities manual.

UF Computing & Networking Services

CNS Information Services

2124 NE Waldo Rd, Suite 2100
Gainesville Florida 32609-8922
(352) 392.2061
<editor@cns.ufl.edu>
# Table of Contents

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARDDISK</td>
<td>1</td>
</tr>
<tr>
<td>JCL</td>
<td>1</td>
</tr>
<tr>
<td>Examples</td>
<td>1</td>
</tr>
<tr>
<td>Cardlist</td>
<td>3</td>
</tr>
<tr>
<td>JCL for Using Cardlist Under CLASS=U</td>
<td>3</td>
</tr>
<tr>
<td>List Control Statements</td>
<td>3</td>
</tr>
<tr>
<td>Standard Batch Version of Cardlist</td>
<td>5</td>
</tr>
<tr>
<td>Tape Handling</td>
<td>6</td>
</tr>
<tr>
<td>Delete</td>
<td>9</td>
</tr>
<tr>
<td>JCL</td>
<td>9</td>
</tr>
<tr>
<td>Example</td>
<td>9</td>
</tr>
<tr>
<td>Dfheofmt</td>
<td>11</td>
</tr>
<tr>
<td>Examples</td>
<td>11</td>
</tr>
<tr>
<td>Completion Codes</td>
<td>12</td>
</tr>
<tr>
<td>Dfhmdset</td>
<td>13</td>
</tr>
<tr>
<td>Ddname</td>
<td>13</td>
</tr>
<tr>
<td>Example</td>
<td>13</td>
</tr>
<tr>
<td>Dsncheck</td>
<td>15</td>
</tr>
<tr>
<td>Jcl</td>
<td>15</td>
</tr>
<tr>
<td>Options Command</td>
<td>15</td>
</tr>
<tr>
<td>Include and Exclude Commands</td>
<td>16</td>
</tr>
<tr>
<td>Title Command</td>
<td>17</td>
</tr>
<tr>
<td>Selecting Direct-Access Volumes</td>
<td>17</td>
</tr>
<tr>
<td>Dsat</td>
<td>19</td>
</tr>
<tr>
<td>Jcl</td>
<td>19</td>
</tr>
<tr>
<td>Accessing Documentation of DSAT Keywords</td>
<td>20</td>
</tr>
<tr>
<td>Iutprogm</td>
<td>21</td>
</tr>
<tr>
<td>Jcl for Using Iutprogm</td>
<td>21</td>
</tr>
<tr>
<td>Command Input</td>
<td>22</td>
</tr>
<tr>
<td>Examples</td>
<td>25</td>
</tr>
<tr>
<td>Documentation</td>
<td>25</td>
</tr>
<tr>
<td>Repro Command of IDCAMS</td>
<td>27</td>
</tr>
<tr>
<td>Copying A Data Set</td>
<td>27</td>
</tr>
<tr>
<td>The Repro Command</td>
<td>27</td>
</tr>
<tr>
<td>Jcl for Using IDCAMS and Repro</td>
<td>27</td>
</tr>
<tr>
<td>Repro Examples</td>
<td>28</td>
</tr>
<tr>
<td>Optional Keywords for the Repro Command</td>
<td>29</td>
</tr>
<tr>
<td>Documentation</td>
<td>29</td>
</tr>
<tr>
<td>Utilities for z/OS (OS/390) Disk Data Sets</td>
<td>iii</td>
</tr>
</tbody>
</table>
List of Figures

1. JCL for the CARDDISK Procedure ................................................................. 1
2. JCL for the CARDLIST Utility in CLASS=U .................................................... 3
3. JCL to Use CARDLIST in CLASSes Other Than U ........................................ 5
4. JCL for Using the DELETE Procedure ............................................................. 9
5. Using DELETE to Delete a Data Set .................................................................... 9
6. JCL for DFHEOFMT .......................................................................................... 11
7. Specifying Multiple Data Sets for DFHEOFMT .............................................. 11
8. JCL for DFHMDSET .......................................................................................... 13
9. JCL for Using the DSNCHECK Procedure ...................................................... 15
10. Example Using the INCLUDE Command of DSNCHECK ............................... 16
11. Adding a Volume Name With DSNCHECK .................................................... 17
12. JCL for Using the DSAT Procedure ................................................................. 20
13. JCL for DSAT Keywords ................................................................................. 20
14. JCL for Using IUTPROGM .............................................................................. 21
15. IUTPROGM Example Using Partitioned Data Sets ........................................ 25
16. IUTPROGM Example Using Sequential Data Sets ........................................ 25
17. JCL to Use IDCAMS and the REPRO Command ........................................... 27
18. Using REPRO to Copy a VSAM Data Set to A Sequential Data Set ................ 28
19. Using REPRO to Backup a VSAM Data Set .................................................... 28
20. JCL to Use IDCAMS and the PRINT Command ............................................. 31
List of Examples

1. Sample of Listing Card Images from a Tape .................................................................7
CARDDISK

You can use the CARDDISK utility to create a disk data set from card images (80-byte records) or to add card images to the end of an existing disk data set. The only required parameter is the data set name to be given to the new disk data set. By default, this will provide a primary allocation of 10 tracks and secondary allocations of 5 tracks. If it is created successfully, the new data set will be cataloged. The record format will be FB (fixed block), with 80 characters per record and a blocksize of 6160. NOTE: The data set name must conform to standard CNS disk data set naming conventions or it will be deleted at the end of the day.

JCL

Use the following JCL for CARDDISK:

Figure 1. JCL for the CARDDISK Procedure

```
//jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
// EXEC CARDDISK,DDSN='data.set.name'
//GO.CARDS DD * ...card images...
```

To override any of the default values, specify any of the optional parameters and values below on the EXEC statement. Commas must separate each parameter.

- `PRIM=#tracks`: specifies the number of tracks needed in primary space allocation. Default is 10.
- `SEC=#tracks`: specifies the number of tracks in secondary space allocation. Default is 5.
- `N1DISP=disp`: specifies the disposition of the new data set at the start of the job step. Default is NEW.
- `DRECFM=format`: specifies output record format. Default is FB.
- `DLRECL=length`: specifies logical record length. Default is 80.
- `DBLKS=size`: specifies block size. Default is 6160.
- `N2DISP=disp`: specifies disposition of disk data set at the end of the job step. Default is CATLG.
- `UNIT=unit`: specifies a type of disk unit. Default is SYSDA.
- `DVOL=volume`: gives the name of a specific disk volume. Default is null.

Examples
The following **EXEC** statement creates a sequential disk data set called 'UF.KITTIES.CATNIP' using default values:

```
// EXEC CARDDISK, DSN='UF.KITTIES.CATNIP'
```

The following **EXEC** statement creates a disk data set with a primary allocation of 50 tracks and no secondary allocation:

```
// EXEC CARDDISK, DSN='UF.PUPPIES.CUDDLE', PRIM=50, SEC=
```

The following **EXEC** statement adds card images to the end of an existing disk data set called 'UF.ISTUDY.BOOKS':

```
// EXEC CARDDISK, DSN='UF.ISTUDY.BOOKS', N1DISP=MOD
```
The CARDLIST utility lets you print a card image file (80-byte records). The CARDLIST utility is available both under CLASS=U and standard batch classes (A, 2, 1, etc.). Under CLASS=U, you can use CARDLIST to generate printed listings of card image files. They may not contain any JCL or JES2 statements. Under standard batch classes, you can control the format of the listing, merge statement and tape input, or include JCL in the statements to be listed.

**JCL for Using CARDLIST Under CLASS=U**

Figure 2. JCL for the CARDLIST Utility in CLASS=U

```plaintext
// jobname JOB ,'your name',CLASS=U,TIME=(mm,ss),LINES=lines<p>/*ROUTE PRINT node.location
// EXEC CARDLIST
...LIST control statements and card images to be listed...
```

Under CLASS=U, the card images to be listed must not contain any JCL statements (lines that begin with // or /* in columns 1 and 2).

**List Control Statements**

The LIST control statements can be continued on more than one line. The following restrictions apply:

- The first statement of a control statement set must contain the phrase $PARMS starting in column 1. $PARMS must be followed by at least one <blank>.
- Only columns 1-72 of the input control statement will be scanned. Characters in columns 73-80 are ignored but can be used for sequence numbering or other identification. The entire statement, however, will be reproduced in the output listing.
- If the control information is to be continued on the next line, each statement to be continued must end in a comma.
- Control information is presented through keyword parameters. The keyword parameter must be coded exactly as spelled in the following sections.
- The keyword parameters must be separated from one another by a comma or at least one blank.
- The operand (information associated with the keyword) must immediately follow the equal sign. If there are any intervening blanks or commas, a syntax error will occur.

Every time a LIST control statement set is encountered in the input stream, the listing currently being produced will be terminated and a new listing will be started if there are more
card images in the input stream. Before the next control statement set is processed, all the control information will be reinitialized to default settings. The default settings are indicated for each control statement in the section below.

A list control statement set with no keyword parameters will terminate the current listing and start a new listing using default control settings.

**Keyword Parameters**

<table>
<thead>
<tr>
<th><strong>TYPE=</strong></th>
<th>specifies the type of listing that is to be produced. Possible operands are as follows:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>80/80</strong></td>
<td>(Default) produces a card image 80/80 listing</td>
</tr>
<tr>
<td><strong>IEBUPDTE</strong></td>
<td>starts a new page every time an <strong>ADD</strong>, <strong>REPL</strong>, or <strong>CHANGE IEBUPDTE</strong> control statement is encountered.</td>
</tr>
<tr>
<td><strong>ASSEMBLER</strong></td>
<td>produces a pseudo-assembler listing (<strong>SPACE</strong> and <strong>EJECT</strong> statements will be simulated). This operand can be abbreviated as <strong>ASM</strong>.</td>
</tr>
<tr>
<td><strong>PL1</strong></td>
<td>produces an 80/80 listing using ASA standard carriage control characters specified in column one of the statement</td>
</tr>
<tr>
<td><strong>FORTRAN</strong></td>
<td>produces a listing with every subroutine or function subprogram starting on a new page. This operand can be abbreviated as <strong>FORT</strong>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>SPACE=</strong></th>
<th>specifies the carriage control that is to be used for this listing. Possible operands are</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SINGLE</strong></td>
<td>(Default) single spacing</td>
</tr>
<tr>
<td><strong>DOUBLE</strong></td>
<td>double spacing</td>
</tr>
<tr>
<td><strong>TRIPLE</strong></td>
<td>triple spacing</td>
</tr>
</tbody>
</table>

| **CC=** | specifies the column on the statement that is to contain an ASA standard carriage control character. If a carriage control character is present in the specified column, it will override |
the spacing specified by the SPACE keyword parameter. The default is CC=0, or no carriage control.

TITLE=

specifies a character string that is to be used as a title for this listing. If the string contains blanks and/or commas, it must be enclosed in apostrophes ('). If apostrophes enclose the string, they cannot be embedded in the string itself. The string cannot contain more than 120 characters. Shorter titles are preferable because titles that need to be continued on another line will contain the continuation comma. The default heading under which the title will appear is "NERDC -- CARDLIST UTILITY". There is no way to override or remove this default heading.

MARGIN=

specifies the number of spaces contained in the left-hand margin of the listing. This operand must be numeric and the number must be greater than or equal to 0, but less than 50. The default is MARGIN=0, or no left margin.

PAGESIZE=

specifies the number of lines to appear on each page of the listing. For single spacing, this operand must be numeric and the number must be greater than 0, but less than 89. If single-space printing is to be done at 6 lines per inch, the number should be less than 59. The default is PAGESIZE=72.

Control Statement Examples

The following control statement set would generate a pseudo-assembler listing, with 50 lines per page, a 10-column left-hand margin, and each page titled LISTING OF PROGRAM 10.

$PARMS
TYPE=ASSEMBLER,PAGESIZE=50,MARGIN=10, TITLE='LISTING OF PROGRAM 10'

The following control statement set would generate a FORTRAN-type listing. All other control information uses default settings.

$PARMS TYPE=FORTRAN

Standard Batch Version of CARDLIST

A standard batch (not CLASS=U) version of CARDLIST is available for mixing input from statements and tape, and for listing statements that contain JCL and/or JES2 control statements. Use the JCL shown in Figure 3 to access the batch version of CARDLIST.

Figure 3. JCL to Use CARDLIST in CLASSes Other Than U

//jobname JOB , 'your name', CLASS=class, TIME=(mm,ss), LINES=lines /*SETUP TAPEC,n ( ... or TAPE9) /*ROUTE PRINT node.location
Tape Handling

// EXEC PGM=CARDLIST
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD (...see "Tape Handling" below)
//SYSIN DD *
...LIST control statements and card images to be listed...

Use //SYSIN DD * for instream data. If JCL is included in the input stream, use
//SYSIN DD DATA,DLM='xx' where 'xx' is two characters of your choice to be used as
the delimiter in columns 1 and 2 (e.g., $$).

All keywords and options used in the CLASS=U version are available in the standard batch
version. The following extensions support tape input and two different types of JCL listings.

TYPE= specifies when a new page should be started. Possible
operands are as follows:

JCL A new page will be started
for every JOB statement
encountered in the input
stream.

JOBSTEP A new page will be started
for every JOB statement
encountered in the input
stream. If the job has at least
two EXEC statements, a
new page will be started for
the second and successive job
steps.

FILE= specifies the number of the file on tape to be used as input for
this listing. The operand must be numeric and it must be a
valid file number. File=0 implies that the SYSUT1 DD
statement completely specifies all the parameters needed to
process the data set.

DSN= is needed only when standard label (SL) tapes are being
processed. This operand is a character string that is the data
set name of the file on the tape to be used as input.

Tape Handling

For tape input, you must include a SYSUT1 DD statement; for non-labeled tapes, all
pertinent DCB parameters must be specified. The type of tape label must be shown by either
LABEL=(,SL) or LABEL=(,NL).

Tape input will be used only if the FILE keyword parameter was specified in a LIST
control statement set and a valid file sequence number was specified. For SL tapes, the DSN
keyword parameter must also be specified. If there is an error in a LIST control statement set
that is requesting tape input, the tape input request is cancelled and input is assumed to come
from the SYSIN input stream.
If a LIST control statement set has been processed that specified tape input, the SYSIN input stream will not be read until an end-of-file is encountered for the specified tape file. When a tape end-of-file is encountered, all control information is reset to default specifications before the next statement from the SYSIN input stream is read. If the first statement read from the SYSIN input stream is not a LIST control statement, a new listing will be started using default control information. Otherwise, a new listing will be started using the information from the list control statement set.

Sample JCL to Use CARDLIST with Tapes

Example 1. Sample of Listing Card Images from a Tape

```plaintext
//jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
/*SETUP TAPEC,1
// EXEC PGM=CARDLIST
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD DSN=data.set.name,UNIT=(TAPEC,,DEFER),
// VOL=SER=mytape,LABEL=(,SL),DISP=OLD
//SYSIN DD *
$PARMS DSN=data.set.name,FILE=filenumber
```
DELETE

The **DELETE** cataloged procedure deletes a specified cataloged data set. If the data set exists at the time you run **DELETE**, then it is deleted (scratched) and uncataloged. If it does not exist at the time you run **DELETE**, an error message is printed, and the rest of your job runs normally. **DELETE** always returns condition code 0000, whether or not it finds a data set with the specified name.

If you need to delete a member of a partitioned data set (PDS) rather then the whole PDS, use the **IUTPROGM** utility described in this document.

### 2/26/2005 Update

**IUTPROGM** is a locally written program that is no longer supported. We recommend that IBM's IEHPROGM or IDCAMS utility programs be used instead.

### JCL

Figure 5 shows an easy job setup to delete a data set using **DELETE**.

#### Figure 4. JCL for Using the **DELETE** Procedure

```
//jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
// EXEC DELETE,DSN='your.dataset.name'
```

### Parameters

The **DELETE** procedure has one required parameter, DSN. Use this parameter to specify the name of the data set you want to delete. The name must be in single quotes.

### Example

To delete a data set named UF.userid.DATA.OLD, use the setup in Figure 2.

#### Figure 5. Using **DELETE** to Delete a Data Set

```
//jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
 // EXEC DELETE,DSN='UF.userid.DATA.OLD'
```

### Note

Once **DELETE** is run on a data set, that data set is gone. Data set storage can be expensive, however, so if you really don't need a data set any more, use **DELETE** to get rid of it.
DFHEOFMT

The DFHEOFMT program writes an end-of-file (EOF) on every track in a data set. Any physical sequential (PS) data set can be formatted using this program. One use of DFHEOFMT is to format CICS log files, which are physical sequential data sets.

As each record is written to the data set, a last-record pointer (called a TTR) is updated to indicate the current position in the data set. Thus when the job either ABENDs or closes the data set normally, the TTR will be updated to reflect the current position in the data set. By opening the data set with DISP=MOD, you can continue to write records onto the end of the data set. If there is a system failure, the data set is not properly closed and the TTR is not updated. The TTR will indicate the last record in the data set that was entered when the last close was done. This TTR would be invalid if any new records had been added to the data set.

DFHMDSET can be used to reset the TTR to the proper place in the data set. DFHEOFMT is used to help DFHMDSET determine where the real end of data is in the data set.

DFHEOFMT operates in single-data-set mode or multiple-data-set mode. Single-data-set mode is specified by giving the DDname of the data set to be formatted using the PARM keyword. Multiple-data-set mode is specified by supplying a DDname prefix. The prefix will be extended to eight characters by concatenation of digits. The numbers appended will be incremented by one until a DD statement cannot be found. For example, if the DDname prefix is 'FVS', the DD statements FVS00001, FVS00002, etc., will be assumed. Examples of PARM keywords for single and multiple data sets follow.

\[ \text{PARM}=\text{'ddname'} \quad \text{Single} \quad \text{-- formats one data set} \]
\[ \text{PARM}=\text{'*ddpre'} \quad \text{Multiple} \quad \text{-- formats more than one data set} \]

Examples

Figure 1 shows an example to format the data set specified by the DD statement TARGET.

**Figure 6. JCL for DFHEOFMT**

```plaintext
//jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
 // EXEC PGM=DFHEOFMT,PARM='TARGET'
 //TARGET DD DSN=data.set.name,DISP=OLD
```

Figure 2 shows an example to format all data sets specified by DD statements with the prefix TARGET.

**Figure 7. Specifying Multiple Data Sets for DFHEOFMT**

```plaintext
//jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
 // EXEC PGM=DFHEOFMT,PARM='*TARGET'
```
Completion Codes

Error conditions are indicated by condition codes other than zero as follows:

<table>
<thead>
<tr>
<th>Completion Code</th>
<th>ABEND Action</th>
<th>Single DS Mode</th>
<th>Multiple DS Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>NODUMP</td>
<td>DDNAME NOT GIVEN</td>
<td>DDNAME PREFIX NOT GIVEN</td>
</tr>
<tr>
<td>8</td>
<td>NODUMP</td>
<td>DD CARD MISSING</td>
<td>N/A (MAPS TO 0)</td>
</tr>
<tr>
<td>12</td>
<td>NODUMP</td>
<td>DD DUMMY OR NONDASD</td>
<td>DD DUMMY OR NONDASD</td>
</tr>
<tr>
<td>16</td>
<td>NODUMP</td>
<td>DSORG NOT PS</td>
<td>DSORG NOT PS</td>
</tr>
<tr>
<td>20</td>
<td>DUMP</td>
<td>OPEN FAILED</td>
<td>OPEN FAILED</td>
</tr>
<tr>
<td>24</td>
<td>DUMP</td>
<td>I/O ERROR</td>
<td>I/O ERROR</td>
</tr>
</tbody>
</table>

You can use the keyword ABEND as the first character in the PARM field so that an error condition can be translated into a user ABEND of the same value. The column labeled ABEND action describes whether or not a particular error condition will be accompanied by a dump. Each error condition will be accompanied by an appropriate error message in the JES2 log.

The currently supported DASD device is the IBM 3390.
DFHMDSET

DFHMDSET (normally used in conjunction with DFHEOFMT for CICS data sets) is used to reset the last record pointer (TTR) of a disk data set so that opening the data set with DISP=MOD will allow data to be added to the end of the data set. Running DFHMDSET is only necessary if the TTR becomes invalid. This can occur if there is a system failure while the data set is open. The system keeps a TTR while the data set is open and only updates the TTR when the data set is closed or if the job abnormally ends.

DDNAME

DFHMDSET can be used with a data set whether the TTR is invalid or not. If the TTR is valid, DFHMDSET has no apparent effect.

DFHMDSET locates the end of the data by looking for an EOF (End of File) or a record that does not conform to the DCB attributes of the data set. Using DFHEOFMT to preformat the data set will ensure that DFHMDSET will be able to find the correct end of data. Thus, it is strongly recommended that DFHMDSET be used in conjunction with DFHEOFMT.

DFHMDSET can reset TTRs for up to 999 data sets in one job step. The format of the DDNAME is shown below, where xxx is a 3-digit number. This number can be no lower than 001.

DFHPSxxx

Example

Figure 8. JCL for DFHMDSET

//jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines /*ROUTE PRINT node.location // EXEC PGM=DFHMDSET //DFHPS001 DD DSN=data.set.name,DISP=OLD //DFHPS002 DD DSN=data.set.name,DISP=OLD //DFHPS003 DD DSN=data.set.name,DISP=OLD
### DSNCHECK

You can use **DSNCHECK** for any of the following:

- list attributes of selected data sets on one or more volumes
- list uncataloged, unqualified, or duplicate data sets
- list data sets using less than a specified percentage of the capacity of each track or of total allocated space

### JCL

Figure 10 shows the JCL for using **DSNCHECK**.

**Figure 9. JCL for Using the **DSNCHECK** Procedure**

```jcl
//jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines /*ROUTE PRINT node.location
// EXEC DSNCHECK
// REPORT.SYSIN DD *
...optional commands...
```

### OPTIONS Command

The **OPTIONS** command has the following format:

**OPTIONS FLAGS=(flag(s),...), BLOCKAGE=nn, USAGE=nn**

These are the possible flags:

<table>
<thead>
<tr>
<th>Flags</th>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>NODSLIST</td>
<td>NODS</td>
<td>skip the attribute list</td>
</tr>
<tr>
<td>NODUPLIST</td>
<td>NODU</td>
<td>skip the duplicate list</td>
</tr>
<tr>
<td>NOQLIST</td>
<td>NOQL</td>
<td>skip the unqualified list</td>
</tr>
<tr>
<td>CLIST</td>
<td>C</td>
<td>list uncataloged data sets</td>
</tr>
<tr>
<td>DLIST</td>
<td>DS</td>
<td>list attributes of data sets</td>
</tr>
<tr>
<td>DUPLIST</td>
<td>DU</td>
<td>list duplicate data sets</td>
</tr>
<tr>
<td>QLIST</td>
<td>QL</td>
<td>list unqualified data set</td>
</tr>
</tbody>
</table>

**BLOCKAGE=nn** (optional) lists all selected data sets using less than the specified percentage of the capacity of each track (in which nn
include and EXCLUDE Commands

The **INCLUDE** command has the following format:

```
INCLUDE INDEX=selection-prefix
```

Data sets starting with the given 1- to 44-character string will be included in the selected reports.

The **EXCLUDE** command has the following format:

```
EXCLUDE INDEX=selection-prefix
```

Data sets starting with the given 1- to 44-character string will be excluded from the selected reports.

The selection-prefix character strings are used in character comparisons. "INDEX=U" and "INDEX=U." will have different results (INDEX=U includes data sets beginning with UF but INDEX=U. does not). You can use as many **INCLUDE** and **EXCLUDE** commands as necessary. In general, **EXCLUDE** commands should specify more detail than **INCLUDE** commands.

For example, the commands below yield all data sets starting with "UF." except for those starting with "UF.A".

```
INCLUDE INDEX=UF. EXCLUDE INDEX=UF.A
```

The following sequence, however, will result in all data sets being excluded:

```
INCLUDE INDEX=UF.A EXCLUDE INDEX=UF.
```

Example of Using **INCLUDE**

Figure 11 shows an example of using the **INCLUDE** command. This example would list all data sets belonging to userid KITTIES that start with either "U." or "UF."

```
// jobname
JOB , 'your name', CLASS=class, TIME=(mm,ss), LINES=lines
/*ROUTE PRINT node.location
// EXEC DSNCHECK
// REPORT.SYSIN DD *
INCLUDE INDEX=U.KITTIES
```

Figure 10. Example Using the **INCLUDE** Command of DSNCHECK.
TITLE Command

TITLE is an optional page heading requested as follows:

TITLE 'heading'

Selecting Direct-Access Volumes

DSNCHECK contains DD statements for user volumes (USERnn) with DDnames of SYSLIB1, SYSLIB2, ... SYSLIBn. You can include additional volumes by supplying additional DD statements. For example, to add a volume named PRU031 use the following:

```
//jobname JOB , 'your name', CLASS=class, TIME=(mm, ss), LINES=lines
// EXEC DSNCHECK
// VTOC.SYSLIBE DD UNIT=SYSDA, VOL=SER=PRU031, DISP=OLD
```

You can override the default volumes by supplying other volume specifications, or you can delete a volume by using a DD DUMMY parameter. For example, to remove the second user volume from consideration, you could use the following:

```
// EXEC DSNCHECK
// VTOC.SYSLIB2 DD DUMMY
```

DSNCHECK uses the following SYSLIB definitions:

<table>
<thead>
<tr>
<th>DDname</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSLIB1</td>
<td>USER90</td>
</tr>
<tr>
<td>SYSLIB2</td>
<td>USER91</td>
</tr>
<tr>
<td>SYSLIB3</td>
<td>USER92</td>
</tr>
<tr>
<td>SYSLIB4</td>
<td>USER93</td>
</tr>
</tbody>
</table>

Note

The DDnames for all volumes to be processed must be unique and must begin with the six characters "SYSLIB". DSNCHECK requires a separate DD statement for each volume the utility is to inspect.
**DSAT**

You can use **DSAT** for displaying allocation information about data sets. **DSAT** will search the catalog for the entries for the data sets specified. Allocation information will be obtained from the volume table of contents (VTOC), formatted, and displayed. If a name is an index name, all data sets below the index will be displayed.

The user may bypass the catalog search by supplying the volume serial on which the data set resides. This option permits displaying information for an uncataloged data set.

The user can select the attributes to be displayed by specifying keyword operands.

The **DSAT** command may be used in TSO CLIST or REXX command procedures to find the allocation of a data set or a group of data sets and set the return code to the specified value. The return code may then be tested with the **WHEN** command. Output may be supressed by specifying **NOPRINT**. The **HARDCOPY** keyword may be used to direct the output to a pre-allocated data set.

The user may choose the information to be displayed by entering keywords. The information that may be displayed is:

1. Volume Serial on which the data set is located
2. File sequence number
3. Device type code from catalog entry
4. Allocation (allocated, used, and extents)
5. Secondary allocation (amount and units)
6. Data set organization
7. **DCB** (**RECFM**, **BLKSIZE**, and **LRECL**).
8. Creation date
9. Expiration date
10. Last date referenced (MVS SU 60)
11. Fully qualified data set name
12. **CCHHR** of the format 1 DSCB
13. Generation Data group data
14. PDS directory information.

**JCL**

---

Utilities for z/OS (OS/390) Disk
Data Sets
Figure 12. JCL for Using the DSAT Procedure

```
// jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
//*ROUTE PRINT node.location
// EXEC TSO
//SYSIN DD *
   DSAT 'UF.userid' PDS SEC LAST GDG
/*
```

Accessing Documentation of DSAT Keywords

To output a list of DSAT keywords, enter the following JCL. Figure 2 shows the JCL for outputting a list of DSAT keywords.

Figure 13. JCL for DSAT Keywords

```
// jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
//*ROUTE PRINT node.location
// EXEC TSO
// SYSIN DD *
   HELP DSAT
/*
```
No Longer Supported

IUTPROGM is a locally written program that is no longer supported. We recommend that IBM's IEHPROGM or IDCAMS utility programs be used instead.

IUTPROGM

IUTPROGM allows you to do the following:

- allocate, scratch (delete), rename, catalog and uncatalog data sets
- create a partitioned dataset member alias
- display DSCB information
- list the space remaining on any online volume

IUTPROGM performs many of the same functions as IBM's IEHPROGM utility. IUTPROGM's syntax, however, is easier to use than IEHPROGM's. Unless otherwise noted, the keywords and operands described here for IUTPROGM have the same meaning as they do in the IBM JCL manuals or for IEHPROGM in the IBM utilities manual.

IUTPROGM cannot be used to create generation data groups. GDGs must be defined using IBM's IDCAMS utility.

JCL for Using IUTPROGM

Figure 14. JCL for Using IUTPROGM

//jobname JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
//GO EXEC PGM=IUTPROGM
//SYSPRINT DD SYSOUT=A
...additional DD statements for PDS members...
//SYSIN DD *
...input statements...

Additional DD Statements for PDS Members

If you want to process the members of more than one partitioned data set, you need to include an additional DD statement for each volume. If the data sets all reside on the same volume, you need to include only one additional DD statement with VOL=SER=. These additional DD statements are not required if processing does not specify PDS members.

If the data set is cataloged, you can use the data set name rather than the volume name; IUTPROGM will look the volume up in the system catalog, as in this example:

//SYSPRINT DD SYSOUT=A
//VOL DD DSN=U.userid.PROJECT,DISP=SHR
Command Input

Commands can appear anywhere between columns 1 and 72 on the input statement. Continuation to another line is allowed, and is indicated by a trailing comma on the first line. Any unique abbreviation can be used for the commands and keywords. (As stated above, unless otherwise noted, the keywords have the same meaning as they do in the IBM JCL manuals or for IEHPROGM in the IBM utilities manual.)

IUTPROGM allows comment statements, indicated by an asterisk (*) in column one. Comment statements can be placed anywhere within the SYSIN data stream, even between successive lines of multi-line commands.

Commands and Operands

Required operands are indicated. Operands not indicated as required are optional. Abbreviations in parentheses indicate acceptable values for the operands.

ALLOCATE

DSNAME= (Required) data set name
VOLSER= (Required) volume on which to place data set
DSORG= data set organization (IS, PS, DA, PO, ISU, PSU, DAU, POU, NONE)
BLKSIZE= blocksize
LRECL= logical record length
RECFM= record format (U, UA, UM, F, FB, FBS, V, VB, VBS, VA, FA, FBA, FBSA, VB, VBSA, FM, FBM, FBSM, VM, VBM, VBSM, VS, FT, FS, NONE)
SPACE= (Required) space allocation (TRK, ABSTR, BLK, CYL)

Note

Only one operand can be specified with the OPTCD= command when allocating a data set.

RKP= record key position
KEYLE= key length
ELNM= element name (PRIME, INDEX, OVERFLOW)
NTM=

number of tracks in master index

SCATCH

DSNAME= (Required) data set name

VOLSER= volume on which data set resides (if omitted, volume information will be obtained from the system catalog)

Note

Scratching a data set does not uncatalog it. You must also use the UNCATLG command to remove the name from the system catalog.

MEMBER= member of data set to delete (if omitted, the entire data set will be deleted)

RENAME

DSNAME= (Required) data set name

NEWNAME= (Required) new name for data set or member

VOLSER= volume on which data set resides (if omitted, the information will be obtained from the system catalog)

MEMBER= name of member to be renamed (if omitted, the entire data set will be renamed)

Note

A data set is not automatically recataloged when it is renamed. You must use the CATLG command to catalog a renamed data set.

ALIAS

DSNAME= (Required) data set name

NEWNAME= (Required) alias for member

MEMBER= (Required) member name to be given an alias

VOLSER= volume on which data set resides (if omitted, the information will be obtained from the system catalog)

CATLG

DSNAME= (Required) data set name
**UNCATLG**

- **DSNAME**= (Required) data set to uncatalog

  **Note**

  Uncataloging a data set does not delete it.

**DISPLAY**

- **DSNAME**= (Required) data set name
- **VOLSER**= volume on which data set resides (if omitted, volume information will be obtained from the system catalog)
- **OPTION**= `OPT=EXTENT` generates extent information

**SEARCH**

- **DSNAME**= (Required) data set to look for

**LSPACE**

- **VOLSER**= list free space on specific volume
- **PREFIX**= list free space on all online volumes beginning with this prefix
- **OPTIONS**= `OPT=ALL` lists free space on all online volumes

**OPTIONS**

- **OPTIONS**= new options for this session. If none are specified, the default values will be restored. The only option currently supported is `OPT=NOECHO` to suppress echoing of input commands to `SYSPRINT`.

**CC**

- **(number)**= (Required) set current completion code.
Return Codes

0  Normal completion
8  At least one error was encountered

Note

If you do not supply a volser to be used, when IUTPROGM reads the system catalog it will issue a message indicating which volser was obtained.

Examples

Figure 16 shows a sample job setup for IUTPROGM using operands to scratch a PDS member and to rename a member.

Figure 15. IUTPROGM Example Using Partitioned Data Sets

//jobname JOB , 'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
//GO EXEC PGM=IUTPROGM
//SYSPRINT DD SYSOUT=A
//name1   DD DSN=U.userid.MY.LIB
//name2   DD DSN=U.userid.MY.OLD.LIB,DISP=OLD
//SYSIN   DD *
SCRATCH MEMBER=MEMBER1,DSNAME=U.userid.MY.LIB
RENAME   MEMBER=ONE,NEWNAME=TWO,
         DSNAMES=U.userid.MY.OLD.LIB

Figure 17 shows an example of displaying, then scratching a cataloged data set. Note that scratching a data set does NOT uncatalog it, so you must also use the UNCATLG command to remove the name from the system catalog.

Figure 16. IUTPROGM Example Using Sequential Data Sets

//jobname JOB , 'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
//GO EXEC PGM=IUTPROGM
//SYSPRINT DD SYSOUT=A
//SYSIN   DD *
DISPLAY DSNAME=U.userid.MY.NEW.FILE,OPT=EXTENT
SCRATCH DSNAME=U.userid.MY.NEW.FILE
UNCATLG DSNAME=U.userid.MY.NEW.FILE

Refer to the "Commands and Operands" section in this chapter for descriptions of required and optional operands.

Documentation
The CNS General Information: CNS Software and Software References [http://docweb.cns.ufl.edu/docs/d0009/d0009.html] manual (D0009) contains recommended references on IBM JCL and utilities, including information on how to order IBM manuals.
REPRO  Command of IDCAMS

Copying A Data Set

Note

The CFILE utility, which was used to copy sequential and ISAM data sets, is not supported. Neither are ISAM data sets. All users with ISAM data sets should convert them to VSAM.

This document describes how to use IDCAMS/REPRO to copy a data set. It is intended as a replacement for CFILE and is not intended as complete documentation on VSAM data sets or the IDCAMS utility. For detailed information on IDCAMS and using VSAM data sets, see the documentation section below.

IDCAMS is an IBM utility that allows you to create and manipulate VSAM data sets. It has several commands. You can use the REPRO command of IDCAMS to copy VSAM and non-VSAM data sets, VSAM clusters, and alternate indexes.

You cannot use REPRO to copy an entire partitioned data set (PDS) because REPRO does not copy the information in the directories. You can, however, use REPRO to copy individual members.

The IBM IEBCOPY utility should be used to copy an entire PDS.

The REPRO Command

The general form of the REPRO command is as follows:

REPRO parameter parameter -
  parameter -

Parameters may be separated by either commas or blanks. You can continue the REPRO statement by coding a hyphen (-) as the last character of a line. The command and parameters may be typed anywhere between columns 2-72.

JCL for Using IDCAMS and REPRO

Figure 18 shows the JCL for a basic job setup to access IDCAMS. The INFILE parameter names the DD statement for the data set that is to be copied. The OUTFILE parameter names the DD statement describing the output or target data set. You would replace "indd" and "outdd" with DD names of your own choosing.

Figure 17. JCL to Use IDCAMS and the REPRO Command

//COPY JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
// EXEC PGM=IDCAMS
// SYSPRINT DD SYSOUT=A
// indd DD DSN= ... (describes the input data set)
// outdd DD DSN= ... (describes the output data set)
REPRO Examples

Figure 19 shows how to use REPRO to copy a VSAM data set to a sequential data set. You could use this to make a sequential backup copy of your VSAM data set. In this example, assume that your VSAM data set has variable-length records. The maximum record size is 100 bytes and you have embedded keys. When you specify your LRECL and BLKSIZE for your output data set, make sure they are at least 4 bytes longer than the longest record in your VSAM data set.

Figure 18. Using REPRO to Copy a VSAM Data Set to A Sequential Data Set

COPY JOB ,"your name",CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
EXEC PGM=IDCAMS
SYSPRINT DD SYSOUT=A
VSAM DD DSN=UF.userid.VSAM.DATASET,DISP=OLD
SEQ DD DSN=UF.useridSEQ.BACKUP,UNIT=SYSDA,
 SPACE=(TRK,(10,10)),
 DISP=(NEW,CATLG),
 DCB=(LRECL=104,BLKSIZE=1004,RECFM=VB)
SYandin DD *
REPRO -
INFILE(VSAM) -
OUTFILE(SEQ)
*/

Figure 20 shows an example of using REPRO to make a VSAM backup of your original VSAM data set. This example assumes that you have already created the backup data set (using IDCAMS). The OLD DD name defines your original VSAM data set. The NEW DD name defines the output data set.

Figure 19. Using REPRO to Backup a VSAM Data Set

COPY JOB ,"your name",CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
EXEC PGM=IDCAMS
SYSPRINT DD SYSOUT=A
OLD DD DSN=UF.userid.VSAM.DATASET,DISP=OLD
NEW DD DSN=UF.userid.VSAM.BACKUP,DISP=OLD
SYandin DD *
REPRO -
INFILE(OLD) -
OUTFILE(NEW)
*/
Optional Keywords for the REPRO Command

Suppose you want only to make a backup of part of your data set. You can use the \textit{SKIP (n)} and \textit{COUNT (m)} keywords on the \texttt{REPRO} command to delimit the records to be copied. The following form of the \texttt{REPRO} command would copy the input data set beginning with the 100th record and copy 500 records.

\begin{verbatim}
REPRO -
  INFILE(VSAM) -
  OUTFILE(SEQ) -
  SKIP(99) -
  COUNT(500)
\end{verbatim}

Documentation

See the IBM manual \textit{DFSMs/MVS Access Method Services for the Integrated Catalog Facility} for detailed information about using \texttt{IDCAMS}. It also contains descriptions of other \texttt{REPRO} keywords not discussed here.

An excellent, but brief, overview of VSAM data sets and the \texttt{IDCAMS} utility, is in the latest version of the JCL text by Gary DeWard Brown.

See the \textit{CNS General Information: CNS Software and Software References} manual (D0009) [http://docweb.cns.ufl.edu/docs/d0009/d0009.html] for information on how to obtain these manuals.
PRINT Command of IDCAMS

Printing a Data Set in Hexadecimal Format

The **Vdump** utility, no longer supported, allowed you to dump a sequential or ISAM data set in EBCDIC followed by the vertical hexadecimal equivalent. ISAM data sets are also no longer supported. All users with ISAM data sets should convert them to VSAM.

This information describes how to use the **PRINT** command of **IDCAMS** to print a sequential or VSAM data set in character and/or hexadecimal format. It is intended as a replacement for **Vdump** and is not intended to describe how to use VSAM data sets. For detailed information on **IDCAMS** and using VSAM data sets, see the documentation section below.

**IDCAMS** is an IBM utility that allows you to create and manipulate VSAM data sets. It has several commands. You can use the **PRINT** command of **IDCAMS** to print a VSAM data set.

The **PRINT** Command

The general form of the **PRINT** command is as follows:

```
PRINT parameter parameter -
    parameter -
    parameter
```

Parameters may be separated by either commas or blanks. You can continue the **PRINT** statement by coding a hyphen (-) as the last character of a line. The command and parameters may be typed anywhere between columns 2-72.

JCL for Using **IDCAMS** and **PRINT**

Figure 21 shows the JCL for a basic job setup to access **IDCAMS**. Replace "format" with **CHARACTER** if you want the data set to be printed in character format, **DUMP** to specify both hexadecimal and character format, or **HEX** to print just in hexadecimal.

**Figure 20. JCL to Use **IDCAMS** and the **PRINT** Command**

```
//PRINT JOB ,'your name',CLASS=class,TIME=(mm,ss),LINES=lines
/*ROUTE PRINT node.location
// EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//indd DD DSN=... (describes the input data set)
//SYSIN DD *
PRINT -
    INFILE(indd) -
    format
/*
```
Optional Keywords

You may want only to print part of your data set. You can use the \textit{\texttt{SKIP(n)}} keyword to indicate the number of records to skip before the printing begins. The following example would begin printing with the 100th record.

\begin{verbatim}
PRINT -
  INFILE(OLD) -
  CHARACTER -
  SKIP(99)
\end{verbatim}

Documentation

See the IBM manual \textit{DFSMS/MVS Access Method Services for the Integrated Catalog Facility} for detailed information about using \texttt{IDCAMS}. It also contains descriptions of other \texttt{PRINT} keywords not discussed here.

An excellent, but brief, overview of VSAM data sets and the \texttt{IDCAMS} utility, is in the latest version of the JCL text by Gary DeWard Brown.

See the \textit{CNS General Information: CNS Software and Software References} manual (D0009) [http://docweb.cns.ufl.edu/docs/d0009/d0009.html] for information on how to obtain these manuals.
Index

D
  DASD, 1