IBM

eserver

iSeries

CL Commands Volume 13

iSeries

CL Commands Volume 13

Contents

Command Descriptions

DSPUSRPRTI (Display User Print Information) Command Description

DSPUSRPRTI Command syntax diagram

Purpose

The Display User Print Information (DSPUSRPRTI) command displays the user print information for the specified user profile.

Optional Parameters

USER Specifies the name of the user whose print information is displayed.

***CURRENT:** The user profile that is currently running is used.

user-name: Specify the name of the user whose user print information is displayed.

OUTPUT
Specifies whether the output from the command is shown at the requesting workstation or printed with the job's spooled output. More information on this parameter is in commonly used parameters.

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

Example for DSPUSRPRTI

```
DSPUSRPRTI   USER(FEIST)
```

This command displays the user print information for user profile FEIST.

Error messages for DSPUSRPRTI

***ESCAPE Messages**

CPF2204
User profile &1 not found.

CPF2217
Not authorized to user profile &1.

CPF2247
Internal security object not available. Reason code &1.

DSPUSRPRF (Display User Profile) Command Description

DSPUSRPRF Command syntax diagram

Purpose

The Display User Profile (DSPUSRPRF) command displays the contents of a user profile. The user profile contains the user's operational limits for system resources, the names of the objects, commands, and devices that the user has specific authority to use, and the names of the objects that the user owns, and that the user is the primary group for.

Objects owned by the user profile are not shown on the *CMDAUT, *DEVAUT, *OBJAUT, or *OBJPGP displays.

This command does not show the password, nor does it show information about objects authorized for public use. The document password is not shown on the *BASIC display or on any CL command output. Any user on the system can be authorized to use the DSPUSRPRF command, but the requesting user must have read authority for the user profile being displayed.

The DSPUSRPRF function may be a long-running function, depending upon the number of objects the user profile owns and is authorized to use.

Restriction: The user name can be specified as USRPRF(*ALL) or USRPRF(generic*-user-name) only when TYPE(*BASIC) and OUTPUT(*OUTFILE) are specified.

Required Parameter

USRPRF

Specifies the name of the user profile.

***ALL:** All user profiles are displayed.

generic-user-name:* Specify the generic name of the user. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic names, refer to generic names.

user-name: Specify the name of the user profile to be displayed.

Optional Parameters

TYPE Specifies the types of user profile information that can be displayed. All, or one, of the following can be displayed:

- The basic portion of the user profile that describes the user
- Commands for which the user profile has specific authority
- Devices for which the user profile has specific authority
- All objects (including commands and devices) for which the user has some specific authority and the authorities assigned with those objects
- Objects that are owned by the user
- Objects that the user is the primary group for.
- Members of the group, if the user profile is a group profile

***BASIC:** All parameters as defined in the user profile are displayed.

***ALL:** All of the information in the user profile is displayed.

***CMDAUT:** The control language (CL) commands, to which the user has specific authority, are displayed.

***DEVAUT:** The system devices to which the user has specific authority are displayed.

***OBJAUT:** The total number and the names of the objects to which the user has specific authority (except those authorized for public use), the user's authority for those objects, and the object types are displayed. Commands and devices are included if *OBJAUT is specified.

***OBJOWN:** For each object owned by the user operating under the user profile, the total number of owned objects and the object names, the types, and the libraries in which the objects reside are displayed.

***OBJPGP:** Displays the total number of objects the user is the primary group for, the object names, the type, the library the object resides in, and the primary group authority.

***GRPMBR:** The members of a group are displayed. This display is available only if the user profile being displayed is a group profile.

OUTPUT

Specifies whether the output from the command is shown at the requesting workstation or printed with the job's spooled output. More information on this parameter is in Commonly used parameters.

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

OUTFILE

Specifies the qualified name of the database file to which the output of the display is directed. If the file does not exist, this command creates a database file in the specified library. If this function creates the file, the text reads 'Outfile for DSPUSRPRF', and the public authority is *EXCLUDE.

Note:

If a new file is created and *BASIC is specified on the **Type of information** prompt (TYPE parameter) the system uses QADSPUPB in QSYS with a format name QSYDSUPB as a model.

If a new file is created and *OBJAUT is specified on the **Type of information** prompt (TYPE parameter) the system uses QADSPUPA in QSYS with a format name QSYDSUPA as a model.

If a new file is created and *OBJOWN is specified on the **Type of information** prompt (TYPE parameter) the system uses QADSPUPO in QSYS with a format name QSYDSUPO as a model.

If a new file is created and *OBJPGP is specified on the **Type of information** prompt (TYPE parameter) the system uses QADSPUPG in QSYS with a format name QSYDSUPG as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file.

OUTMBR

Specifies the name of the database file member to which the output is directed.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter. If the member exists, the user can choose to either add records to the end of the existing member or to clear the existing records in the member and then add the new records.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

Examples for DSPUSRPRF

Example 1: Displaying Basic Information

```
DSPUSRPRF  USRPRF(THSMITH)
```

This command shows the basic portion of the user profile named THSMITH because TYPE(*BASIC) is assumed. The commands, devices, and objects that the user is authorized to use are not displayed. Because OUTPUT(*) is also assumed, the operational information is either displayed or printed, depending on where the command is submitted.

Example 2: Printing a List of Objects

```
DSPUSRPRF  USRPRF(RTJOHNSON)  TYPE(*OBJOWN)
  OUTPUT(*PRINT)
```

This command causes the list of objects that are owned by the user named RTJOHNSON to be printed. The list contains the object names, object types, and the names of the libraries where the objects are located.

Error messages for DSPUSRPRF

*ESCAPE Messages

CPF22DF

Unable to process request for user profile &1.

CPF22D8

Use of generic user profile name not correct.

CPF22D9

No user profiles of specified name exist.

CPF22EB

Unable to process request for user profile &1.

CPF2204

User profile &1 not found.

CPF2213

Not able to allocate user profile &1.

CPF2217

Not authorized to user profile &1.

CPF2257
User profile &1 not a group profile.

CPF9860
Error occurred during output file processing.

DSPUDFS (Display User-Defined File System) Command Description

DSPUDFS Command syntax diagram

Purpose

The Display User-Defined File System (DSPUDFS) command displays the attributes and, optionally, the extended attributes for an existing user-defined file system (UDFS).

Required Parameter

UDFS Specifies the path name of the user-defined file system to be displayed. » It must be in one of the following two forms:

- */dev/QASPXX/udfsname.udfs*, where *XX* is one of the valid system or basic user auxiliary storage pool (ASP) numbers on the system, and *udfsname* is the name of the user-defined file system. All other parts of the name must appear as in the example above.
- */dev/aspname/udfsname.udfs*, where *aspname* is one of the valid independent ASP names on the system, and *udfsname* is the name of the user-defined file system. All other parts of the name must appear as in the example above.

The name part of the path must be unique within the specified *QASPXX* or *aspname* directory. «

Optional Parameter

OUTPUT
Specifies whether the output from the command is shown at the requesting workstation or printed with the job's spooled output. More information on this parameter is in Commonly used parameters.

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output. The output is printed with the job's spooled output.

Example for DSPUDFS

```
DSPUDFS  UDFS('/dev/QASP05/joe.udfs')
```

This command displays the attributes of a user-defined file system (UDFS) named *joe* in the user auxiliary storage pool (ASP) 5.

Error messages for DSPUDFS

None

DSPWSUSR (Display Work Station User) Command Description

DSPWSUSR Command syntax diagram

Purpose

The Display Work Station User (DSPWSUSR) command allows the user to display information describing the current job. Information displayed includes user name or description, system name, work station ID or description, job name, job group, date, and time.

Optional Parameter

OUTPUT

Specifies whether the output from the command is shown at the requesting workstation or printed with the job's spooled output. More information on this parameter is in commonly used parameters.

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

Example for DSPWSUSR

```
DSPWSUSR   OUTPUT(*)
```

This command displays the information describing the current job.

Error messages for DSPWSUSR

None

DO (Do) Command Description

DO Command syntax diagram

Purpose

The Do (DO) command allows the user to group commands within a CL program. It is used with the ENDDO command to identify a group of commands that are run together as a group. Usually, the DO command specifies the starting of a group of commands that are run as a result of a decision made by the processing of an IF command. However, the DO command does not have to be associated with an IF command. When used with an IF command, the DO command can be either the true part of the decision (that is, the value of the THEN parameter of the IF command), or the false part of a decision (on the ELSE command). Every Do group must be ended by the ENDDO command. Do groups can be nested within other Do groups, but each group must have an ENDDO command to end its level of nesting.

Restriction: This command is valid only within a CL program. Up to 10 levels of Do groups can be nested within each other.

There are no parameters for this command.

Examples for DO

Example 1: Processing a Group of Commands

```
DO
   *
   *   (group of CL commands)
   *
ENDDO
```

The commands between the DO and ENDDO commands are processed once, as a group of commands.

Example 2: Processing a Group of Commands

```
If &SWITCH DO
   *
   *   (group of CL commands)
   *
ENDDO
```

The commands between the DO and ENDDO commands are processed if the value in the logical variable &SWITCH is ’1’. If &SWITCH is not ’1’, then control passes immediately to the next command following the ENDDO command.

Error messages for DO

None

DMP (Dump) Command Description

DMP Command syntax diagram

Purpose

The Dump (DMP) command dumps the contents and/or attributes of the specified integrated file system object to a spooled printer file named QPSRVDMP. Whether the contents and/or attributes can be dumped depends upon the object type. Any integrated file system object can be dumped, but only one object can be specified at a time on this command.

Restrictions:

1. The user must have *RX authority to the directory containing the object and *R to the object. If the object is in QSYS.LIB file system, the user must have *USE authority to the object and *EXECUTE authority to the library. To dump internal document library objects *ALLOBJ special authority is required.
2. The Dump (DMP) command will not allow to specify a pattern for a directory, only for the object name. You can invoke the Work with Link (WRKLNK) command to see all the objects you have.
3. For more information on specifying path names, refer to path names. Additional information about object name patterns is in the Integrated File System Introduction topic in the Information Center.
4. Not all file systems support the DMP command, following is a list of local file systems supported:
   - Root file system
   - QOpenSys file system
   - QSYS.LIB file system
   - QDLS file system
   - User defined file systems

Required Parameter

OBJ Specifies the path name of the object to dump.

object-path-name: Specifies the path name of the system object being dumped. If a pattern is specified on this parameter and more than one object matches the pattern, you can select the object from a list in an interactive job. If this is a batch job, the command fails with the error message CPFA08E, “More than one name matches pattern.”

Example for DMP

Example 1: Dumping Stream File Contents

```
DMP   OBJ('/user/Test.stmf')
```

This command dumps the contents of the stream file named /user/Test.stmf. The dump is spooled to the printer output file QPSRVDMP.

DMPBRM (Dump BRM) Command Description

Note: To use this command, you must have the 5722-BR1 (Backup Recovery and Media Services for iSeries) licensed program installed. For detailed information on the parameters of this command, see the online help.

DMPBRM Command syntax diagram

Purpose

The Dump BRM (DMPBRM) command saves BRMS files and related files to assist IBM support personnel in problem determination. You can specify various levels of detail and one or more job logs to dump. This command saves various files from QUSRBRM and related OS/400 files (depending on the specified level) to a device or save file. This information is used in problem determination by your technical representative. Processing this command should be done in conjunction with this representative.

Example for DMPBRM

Example 1: Producing a Level 2 BRMS Dump

```
DMPBRM DEV(TAP01) LVL(2) JOBLOG(*JOB)
```

In this example level 2 dump information along with the job log for the currentjob is to be written to tape unit TAP01.

Error messages for DMPBRM

None »

DMPCLUTRC (Dump Cluster Trace) Command Description

DMPCLUTRC Command syntax diagram

Purpose

The Dump Cluster Trace (DMPCLUTRC) command is used for problem analysis. It dumps cluster-related trace and debug information to a file. The information is dumped locally on one or more cluster nodes, for one or more cluster resource groups. Each cluster resource group that is dumped has a file member in the file. The name of the file member is the name of the cluster resource group. The information dumped is dependent on the particular cluster resource group. The amount of information dumped is determined by the dump level. Only nodes that have an active Cluster Resource Services job for the specified cluster resource group will have a dump output.

Restrictions

1. To use this command you must have either *SERVICE authority or be authorized to the Service Trace function of the operating system through iSeries Navigator’s Application Administration support. You must also have *USE authority to any cluster resource group object that is to be dumped with this command.
2. The cluster must be at version 3 or greater for this command to work remotely (work on any node other than the node issuing the command).
3. Cluster Resource Services must either be active or in the process of starting on the node that this command is issued from.

4. Only nodes that have a job for the desired cluster resource group may particpate in this command.
5. To determine if this command succeeded, check the affected nodes for a dump file. If a file is not there, then check the job log for the associated cluster job for messages.

Required Parameters

CLUSTER

Specifies the name of the cluster for which information is to be dumped or printed.

cluster-name: Specifies the cluster name.

CRG Specifies the name of the cluster resource group that is to be dumped. Possible values are:

***ALL:** All groups, including the reserved names QCSTCTL and QCSTCRGM.

cluster-resource-group-name: Specifies the name of the cluster resource group. The reserved names for the Cluster Control and Cluster Resource Group Manager groups, QCSTCTL and QCSTCRGM, respectively, may also be specified.

Optional Parameters

NODE Specifies the cluster node that is to be dumped. Possible values are:

***LOCAL:** The current node, that is, the node this command is issued on.

***ALL:** All active nodes in the cluster.

node-identifier: Specifies the cluster node identifier.

LEVEL

Specifies the dump level. The amount and kind of information in each level is dependent on the particular cluster resource group being dumped.

Possible values are:

***BASIC:** Specifies the basic level of dump information. This dumps information that is maintained continuously as flight recorder information.

***ERROR:** Specifies the error level of dump information. This dumps error information, and includes the *BASIC level information.

***INFO:** Specifies the informational level of dump information. This dumps completion and warning information, and includes the *ERROR level.

***VERBOSE:** Specifies the verbose level of dump information. This dumps detailed trace and debugging information, and includes the *INFO level.

FILE Specifies the qualified name of the physical file that the dump is written to. The file is written on each node that the dump is requested on in accordance with the OVERWRITE parameter. The same library name is used on all nodes. If a file name is specified other than *NODE, then the file name will also be the same on all nodes.

The name of the physical file is qualified by one of the following values:

QGPL: The file will be created in library QGPL.

***CURLIB:** The current library of the job that is invoking this command is used. The library is determined before a dump request is sent to any other node.

library-name: Specify the name of the library. The same library is used on all specified nodes. No dump is taken on any node that does not have the library.

***NODE:** The cluster node identifier is used as the file name. For example, if a cluster node identifier is NODE1, then the file name is NODE1. If multiple nodes are being dumped, then each node will have a different file name.

file-name: Specify the name of the file. This name is used on all nodes.

OVERWRITE

Specifies whether the specified file will be overwritten or not. If the file exists, it will be deleted and re-created. This parameter is checked on a per node basis. If *NO is specified along with multiple nodes, then only those nodes that do not have the file will have dumps taken. A CPDBB07 message is sent to the job log of every Cluster Resource Services job that participates in the dump that indicates success, failure, or the file cannot be overwritten on the node.

***YES:** The specified file will be overwritten.

***NO:** The specified file will not be overwritten. If the file exists, no dump on the specified node occurs.

Examples for DMPCLUTRC

Example 1: Dumping one cluster resource group on one node

```
DMPCLUTRC CLUSTER(EXAMPLE) CRG(CRG1) NODE(NODE1)
```

This command dumps cluster resource group CRG1 on the node NODE1 in cluster EXAMPLE. On NODE1, a file is created with the name QGPL/NODE1. It has one member named CRG1.

Example 2: Dumping one cluster resource group on all nodes

```
DMPCLUTRC CLUSTER(EXAMPLE) CRG(CRG1) NODE(*ALL) LEVEL(*ERROR) FILE(QGPL/*NODE)
          OVERWRITE(*NO)
```

This command dumps error information from cluster resource group CRG1 on all cluster nodes if the file does not exist. Each node checks individually for the file already existing. The name of the file is QGPL/*node-identifier*.

Example 3: Dumping all cluster resource groups on all nodes

```
DMPCLUTRC CLUSTER(EXAMPLE) CRG(*ALL) NODE(*ALL) LEVEL(*INFO) FILE(*CURLIB/DUMP)
          OVERWRITE(*YES)
```

This command dumps all cluster resource groups on all nodes. The library name is determined by the job that invoked this command. If that library name is MYLIB, then each node has a file named MYLIB/DUMP, with one file member per group dumped in addition to members for QCSTCTL and QCSTCRGM. The file will be destroyed if it exists and re-created for the dump.

Error messages for DMPCLUTRC

***ESCAPE Messages**

CPF0001
Error found on &1 command.

CPF222E
&1 special authority is required.

CPF98A2
Not authorized to &1 command.

CPFBB02
Cluster &1 does not exist.

CPFBB09
Cluster node &1 does not exist in cluster &2.

CPFBB0F
Cluster resource group &1 does not exist in cluster &2.

CPFBB70
Request &1 not compatible with current cluster version.

CPFBBA0
Node &1 in cluster resource group &2 is not responding.

«

DMPCLPGM (Dump Control Language Program) Command Description

DMPCLPGM Command syntax diagram

Purpose

The Dump CL Program (DMPCLPGM) command dumps variables (used in the CL program in which the command is processed) and all messages on the program's message queue to a spooled printer file (QPPGMDMP). This command is valid only in a CL program; after the program is dumped, it continues processing.

Restriction: The user of this command must have read authority for the program.

There are no parameters for this command.

Example for DMPCLPGM

```
PGM
DCL . . .
DCL . . .
MONMSG  MSGID(CPF9999)  EXEC(GOTO DUMP)
  *
  *
  *
RETURN
DUMP:  DMPCLPGM
ENDPGM
```

This CL program monitors for the function check message CPF9999. If a function check occurs in the program, control is passed to the command at label DUMP. This causes a dump of the program's message queue and causes the program's variables to be printed. This dump can be used to determine the cause of the function check.

Error messages for DMPCLPGM

***ESCAPE Messages**

CPF0570
Unable to dump CL program &1 in &2.

DMPDLO (Dump Document Library Object) Command Description

DMPDLO Command syntax diagram

Purpose

The Dump Document Library Object (DMPDLO) command is used primarily for problem analysis. It copies the contents and/or attributes of folders, documents, or internal document library system objects to a spooled printer file named QPSRVDMP. If the printed output is not spooled, and the printer is not available, the printer file (QPSRVDMP) is overridden.

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.
2. The user must have read authority to a document or folder to dump it.
3. The user must have *ALLOBJ authority to dump internal system objects.

Required Parameter

DLO Specifies the document library object being dumped.

***SYSOBJNAM:** A system object name is used to identify the document or folder being dumped. This value must be used to dump an internal or distribution document, or a document that is not in a folder.

***INT:** Internal document library system objects are dumped.

document-or-folder-name: Specify the name of the document or folder being dumped.

Optional Parameters

FLR Specifies the name of the folder that contains the document.

Note: A folder name can be entered in this parameter only if a folder or document name is entered in the DLO parameter.

***NONE:** The object is not in a folder.

folder-name: Specify the name of the folder that contains the folder or document being dumped.

SYSOBJNAM

Specifies the system object name. This parameter is valid only when DLO(*SYSOBJNAM) or DOCL(*SYSOBJNAM) is specified. A full ten characters must be specified.

SYSOBJATR

Specifies attributes of the object being dumped. A value other than *NONE applies only to documents, and can be entered in this parameter only if *SYSOBJNAM is specified on the DLO parameter.

***NONE:** No attributes are specified for the object.

***INTDOC:** The object being dumped is an internal document.

***DST:** The object being dumped is a distribution document.

Examples for DMPDLO

Example 1: Dumping a Document

```
DMPDLO  DLO(KAREN)  FLR(PEGGY)
```

This command dumps a document or a folder named KAREN which is located in the folder named PEGGY.

Example 2: Specifying a System Object Name

```
DMPDLO  DLO(*SYSOBJNAM) SYSOBJNAM(BHZM052634)
```

This command dumps the document library object identified by the system object name BHZM052634.

Error messages for DMPDLO

*ESCAPE Messages

CPF8A43

Dump failed or partially failed for &2 of type &4 in folder path &1.

DMPJVM (Dump Java Virtual Machine) Command Description

DMPJVM Command syntax diagram

Purpose

The Dump Java Virtual Machine (DMPJVM) command dumps information about the Java Virtual Machine (JVM) for a specified job. The information is dumped using printer file QSYSPRT. The user data for the QSYSPRT file is ’DMPJVM’. The dump includes formatted information about the classpath, garbage collection, and threads associated with the JVM.

Restrictions:

- This command uses the Start Service Job (STRSRVJOB) and Start Debug (STRDBG) commands. The user of this command must be authorized to those commands.
- This command is shipped with public *EXECUTE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.
- This command must be run under a user profile that is the same as the job user identity of the JVM job, or that has use (*USE) authority to the job user identity of the JVM job.
- This command is not allowed if the remote service operation has been started for another job and that job is not the same job specified on this command.
- This command is not allowed if the JVM job is held, suspended, or ending.

JOB Specifies the name of the job where the Java Virtual Machine (JVM) is running. If no job number is given, all of the jobs currently in the system are searched for the simple job name. The job name entered must be a job in which a JVM is currently running.

***SRVJOB:** Information about the JVM in the job currently being serviced will be dumped. If no job is currently being serviced, then a job identifier is required.

A job identifier is a qualified name with up to three elements. For example:

```
job-name
user-name/job-name
job-number/user-name/job-name
```

job-name: Specify the name of the JVM job.

user-name: Specify the name of the user of the JVM job.

job-number: Specify the number of the JVM job.

STACKFRAME

Specifies the maximum number of stack frames to be processed for each thread. This value must be greater than zero and cannot be greater than 100. If there are more than the specified number of frames on a thread's stack, the more recent frames on the stack are processed and '...' is used to indicate that not all of the stack frames were processed.

10: A maximum of ten stack frames will be processed for each thread.

***ALL:** All stack frames will be processed for each thread. If a thread has more than 100 stack frames, only the first 100 frames will be processed.

number: The maximum number (1-100) of stack frames that will be processed for each thread.

DUPJOBOPT

Specifies the action taken when duplicate jobs are found by this command.

***SELECT:** The selection display is shown when duplicate jobs are found during an interactive session. Otherwise, an escape message is issued.

***MSG:** An escape message is issued when duplicate jobs are found.

Example for DMPJVM

```
DMPJVM JOB(099246/FRED/QJVACMDSRV)
```

This command dumps the information for the Java Virtual Machine (JVM) running in the job named 099246/FRED/QJVACMDSRV.

Example Output

```
JAVA VIRTUAL MACHINE INFORMATION: 099246/FRED/QJVACMDSRV
 .......................................................................
 . Classpath
 .......................................................................
/QIBM/ProdData/Java400/jdk117/lib/jdkptf117.zip:/QIBM/ProdData/Java400/j
17/lib/classes.zip:/QIBM/ProdData/Java400/ext/IBMmisc.jar:/QIBM/ProdData
va400/jdk117/db2_classes.jar:/QIBM/ProdData/Java400/ext/jssl.jar:/QIBM/P
Data/Java400/ext/ibmjssl.jar:/QIBM/ProdData/Java400/:/fred
 .......................................................................
 . Garbage collection
 .......................................................................
Garbage collector parameters
   Initial size: 2048 K
   Max size: *NOMAX
Current values
   Heap size: 9476 K
   Garbage collections: 0
 .......................................................................
 . Thread information
 .......................................................................
Returned info for 3 thread(s) of 3 thread(s) processed
Thread: Thread-0 TDE-> B000100005B13000
  Thread priority: 5
  Thread status: Running
  Thread group: main
  Runnable: java/lang/Thread @E5DD5798AD001720
 Stack:
    None
 Locks:
      None
 .....................................................................
Thread: t1 TDE-> B000100005B37000
  Thread priority: 5
  Thread status: Java wait
  Thread group: main
  Runnable: dbgtest2 @FABE9958BA001300
 Stack:
    pressEnter.theFirstMethod(Ljava/lang/String;)V+0 (dbgtest2.java:14)
    dbgtest2.run()V+69 (dbgtest2.java:44)
    java/lang/Thread.run()V+11 (Thread.java:466)
 Locks:
      None
 ......................................................................
Thread: t2 TDE-> B000100005B33000
  Thread priority: 5
  Thread status: Timed wait
  Thread group: main
  Runnable: dbgtest2 @FABE9958BA0013C0
 Stack:
    java/io/BufferedInputStream.read()I+11 (BufferedInputStream.java:154)
    pressEnter.theFirstMethod(Ljava/lang/String;)V+10 (dbgtest2.java:15)
    dbgtest2.run()V+69 (dbgtest2.java:44)
    java/lang/Thread.run()V+11 (Thread.java:466)
 Locks:
      None
 ......................................................................
```

Error messages for DMPJVM

***ESCAPE Messages**

JVAB600
Job not found.

JVAB601
DMPJVM failed with reason code &1.

JVAB602
Job parameter required.

JVAB603
Unable to open print file.

DMPJOB (Dump Job) Command Description

DMPJOB Command syntax diagram

Purpose

The Dump Job (DMPJOB) command dumps the basic data structures, or specific calls of the current job or of the job being serviced as a result of the Start Service Job (STRSRVJOB) command. The information is dumped to a spooled printer file (QPSRVDMP) to be printed. If the user had specified SPOOL(*NO) on either the CHGPRTF command or the OVRPRTF command, then the output is not spooled but printed directly; and, if the printer is not available, then this command overrides the print job and spools the output. When the user specifies SPOOL(*NO) on one of the two commands above, the user must specify QPSRVDMP as the printer file. The dump includes formatted information about the specified programs, and dumps of specified OS/400 system objects, system objects, and threads associated with the job.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Required Parameters

PGM Specifies which program to dump. Either a single value or a list of values (10 maximum) can be specified.

***ALL:** All programs on the call stack are dumped.

***NONE:** No programs are dumped. Only the lists of called and activated programs are dumped.

Element 1: Program Name

The name of the dumped program can be qualified by one of the following library values:

***ALL:** All libraries on the system are searched. If *ALL is specified, a call level cannot be specified.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the called program to dump. Up to 10 characters for the program name can be specified.

Element 2: Program Call Levels

***LAST:** The last call with the specified name is dumped.

***FIRST:** The first call with the specified name is dumped.

***ALL:** All calls with the specified name are dumped.

call-level: Specify the level of call for a program with multiple calls in the stack. If *ALL is specified for the library name, the call level cannot be specified.

JOBARA

Specifies whether the job structure areas of the process are dumped. Job structure areas consist of the following:

- Work Control Block
- Library Search List
- Job Temporary Library
- Job Local Data Area
- Spool Control Block
- Data Management Communications Queue
- Service Communication Object
- Process Definition Template
- Process Lock List

- Machine Interface (MI) Response Queue

***ALL:** The job structure areas are dumped.

***NONE:** The job structure areas are not dumped.

ADROBJ

Specifies whether objects addressed from the program storage of a program being dumped are also dumped. If *NONE is specified for the PGM parameter, no addressed objects are dumped.

***YES:** The addressed objects are dumped.

***NO:** The addressed objects are not dumped.

JOBTHD

Specifies whether the list and information of the threads in the job is dumped.

Thread information consist of the following:

- For the thread running the DMPJOB command:
  - Thread Control Block (TCB).
- For all the threads
  - Thread ID
  - Thread handler
  - Thread execution status (hexadecimal value)
  - Thread wait status (hexadecimal value)
  - Thread call stack

***YES:** The thread list and information is dumped.

***NO:** The thread list and information is not dumped.

***THDSTK:** Only the thread call stack is dumped.

SLTTHD

Specifies a list of up to twenty threads in the job whose information is to be included. If *NO is specified on the Job Threads prompt (JOBTHD parameter), no threads are dumped.

***ALL:** All threads are dumped.

***SELECT:** A list of thread identifiers is shown from which the user can select up to twenty to be included.

thread-identifier: Specify the identifiers of up to twenty threads whose information is to be included.

Examples for DMPJOB

Example 1: Dumping Programs

```
DMPJOB   PGM((QGPL/UPDATE *FIRST)
  (PAYROLL/MASTER *ALL))
  JOBARA(*ALL)  ADROBJ(*NO)
```

This command dumps the first occurrence of QGPL/UPDATE in the call stack and all occurrences of PAYROLL/MASTER. The job structure areas are dumped.

Example 2: Dumping Entire Job Structure

```
DMPJOB
```

This command dumps the entire job structure.

Example 3: Dumping Lists of Called and Activated Programs

```
DMPJOB   PGM(*NONE)  JOBARA(*NONE)
```

This command dumps the lists of programs called and activated.

Example 4: Dumping job thread list and information

```
DMPJOB   PGM(*NONE)  JOBARA(*NONE)   JOBTHD(*YES)
```

This command dumps the list of the job's threads and their information.

Example 5: Dumping only one job thread information

```
DMPJOB   PGM(*NONE)  JOBARA(*NONE)
  JOBTHD(*YES) SLTTHD(00000001)
```

This command dumps thread identifier 00000001 and its information.

Example 6: Dumping only the thread call stack

```
DMPJOB   PGM(*NONE)  JOBTHD(*THDSTK)
```

This command dumps only the job's threads call stack.

Error messages for DMPJOB

*ESCAPE Messages

CPF3546
Program parameters specified were not found.

CPF3560
Job being serviced not running.

CPF3563
Overflow value for file &1 in &2 too large.

CPF3585
Library name *ALL and call level cannot be used together.

CPF3909
Service command will not be processed.

CPF3918
Service request canceled.

CPF3925
Cannot open file &1.

CPF3935
Job being serviced ended during dump.

CPF3950
Error message &2 received for file &1. Request ended.

CPF3951
File &1 cannot be overridden by file name &2.

CPF3967
Dump not started because serviced job not running.

CPF3968
Dump not started because serviced job completed running.

CPF3969
Error during close of file &1. Output may not be complete.

DMPJOBINT (Dump Job Internal) Command Description

DMPJOBINT Command syntax diagram

Purpose

The Dump Job Internal (DMPJOBINT) command dumps the machine internal data related to the machine process of the current job or the job being serviced as a result of the Start Service Job (STRSRVJOB) command. This data is for use by IBM service representatives. When the internal data is dumped, a dump identifier is sent in a message to the user who sent the DMPJOBINT command. The Print Internal Data (PRTINTDTA) command can be used to print the dump output.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

There are no parameters for this command.

Example for DMPJOBINT

```
DMPJOBINT
```

This command dumps, for the job in which the command is entered, the machine internal data associated with the job. A message with the dump identifier is sent to the user entering the command.

Error messages for DMPJOBINT

***ESCAPE Messages**

CPF3560
Job being serviced not running.

CPF3636
Internal job not dumped.

CPF3909
Service command will not be processed.

CPF3918
Service request canceled.

CPF3935
Job being serviced ended during dump.

CPF3950
Error message &2 received for file &1. Request ended.

CPF3967
Dump not started because serviced job not running.

CPF3968
Dump not started because serviced job completed running.

DMPOBJ (Dump Object) Command Description

DMPOBJ Command syntax diagram

Purpose

The Dump Object (DMPOBJ) command dumps the contents and/or attributes of the specified OS/400 system object to a spooled printer file named QPSRVDMP. Whether the contents and/or attributes can be dumped depends upon the object type. If the user had specified SPOOL(*NO) on either the CHGPRTF command or the OVRPRTF command, then the output is not spooled but printed directly; and, if the printer is not available, then this command overrides the print job and spools the output. When the user specifies SPOOL(*NO) on one of the two commands above, the user must specify QPSRVDMP as the printer file. Any library or an OS/400 system object that is stored in a library can be dumped, but only one object can be specified at a time on this command.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. The following user profiles have private authorities to use the command:
   - QPGMR
   - QSYSOPR
   - QSRV
   - QSRVBAS

Required Parameters

OBJ Specifies the qualified name of the OS/400 system object being dumped.

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the name of the object being dumped. Only the OS/400 system objects that are stored in libraries can be dumped. Refer to the OBJTYPE parameter for the valid types of objects.

OBJTYPE
Specifies the object type of the OS/400 system object being dumped. Any one of the OS/400 system object types can be specified. More information on this parameter is in commonly used parameters.

Examples for DMPOBJ

Example 1: Dumping File Contents

```
DMPOBJ   OBJ(ORDENT/ORDERIN)  OBJTYPE(*FILE)
```

This command dumps the contents of the file named ORDERIN that is stored in the ORDENT library.

Example 2: Dumping a Program

```
DMPOBJ   OBJ(MYPROG)  OBJTYPE(*PGM)
```

This command dumps the first copy of the program MYPROG that is found in the library list. The dump is spooled to the printer output file QPSRVDMP.

Error messages for DMPOBJ

***ESCAPE Messages**

CPF3560
Job being serviced not running.

CPF3561
Context &8 &9 &7 not found.

CPF3562
Object &7 not found.

CPF3673
Not authorized to library &7.

CPF3909
Service command will not be processed.

CPF3918
Service request canceled.

CPF3925
Cannot open file &1.

CPF3935
Job being serviced ended during dump.

CPF3946
Context damaged.

CPF3947
Library &7 not available.

CPF3948
Library &3 previously deleted.

CPF3949
Library &7 damaged.

CPF3950
Error message &2 received for file &1. Request ended.

CPF3951
File &1 cannot be overridden by file name &2.

CPF3967
Dump not started because serviced job not running.

CPF3968
Dump not started because serviced job completed running.

CPF3969
Error during close of file &1. Output may not be complete.

DMPSYSOBJ (Dump System Object) Command Description

DMPSYSOBJ Command syntax diagram

Purpose

The Dump System Object (DMPSYSOBJ) command is used primarily for problem analysis. It dumps the contents and/or attributes of machine interface (MI) system objects to a spooled printer file named QPSRVDMP. If the user had specified SPOOL(*NO) on either the CHGPRTF command or the OVRPRTF command, then the output is not spooled but printed directly; and, if the printer is not available, then this command overrides the print job and spools the output. When the user specifies SPOOL(*NO) on one of the two commands above, the user must specify QPSRVDMP as the printer file. Any MI object that is stored in any library (context) or that is addressable through an object can be dumped. A specific object, a generic group, or all of the MI objects in a context can be specified. The dump operation can also be limited to objects of a specified type and, optionally, of a specified subtype.

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.
2. The user must have use authority to the object and read authority to the program and the user profile.

Optional Parameters

OBJ Specifies which of the MI system objects are being dumped. The name of a specific object, the generic name of a group of objects, the process control space of the job, the machine context, or all of the MI objects in a context can be specified. If a library name is specified, the library is dumped, but not the objects in it.

If OBJ(QTEMP) is specified along with either OBJTYPE(*LIB) or TYPE(04) SUBTYPE(01), the temporary job context associated with the job that this command is entered from, or the job being serviced as a result of the Start Service Job (STRSRVJOB) command, is dumped. In either case, the CONTEXT parameter is ignored.

***PCS:** The process control space of the current job or that of the job being serviced as a result of the Start Service Job (STRSRVJOB) command is dumped. OBJ(*PCS) can be used with the OFFSET and SPACE parameters to dump objects in the job structure. If OBJ(*PCS) is specified, the CONTEXT, TYPE, SUBTYPE, and OBJTYPE parameters are ignored.

***MCHCTX:** The machine context (which contains a list of the objects in the context) is dumped. If OBJ(*MCHCTX) is specified, all the other parameters in this command are ignored.

***ALL:** All the MI system objects in the specified context are dumped if they match the requirements specified in TYPE and SUBTYPE (for MI objects), or OBJTYPE (for the OS/400 system objects).

generic-system-object-name:* Specify the OS/400 system or MI generic object name that identifies the group of MI system objects to be dumped. An MI object name can have up to 30 characters in it. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic names, refer to generic names.

system-object-name: Specify the name of the OS/400 system or MI object that is to be dumped. Up to 30 characters can be entered. If more than one object has the same name, all objects having that name and matching the attributes specified by the CONTEXT parameter, and either the TYPE and SUBTYPE parameters or the OBJTYPE parameter are dumped. If a specific object is being dumped, the CONTEXT, TYPE, and SUBTYPE parameters or the CONTEXT and OBJTYPE parameters should be specified.

CONTEXT

Specifies which context or library the objects to be dumped are found in.

***NONE:** The object specified by the OBJ parameter is not in any context. *NONE is valid only if *PCS or *MCHCTX is specified or assumed for the OBJ parameter, or if OBJ(QTEMP) is specified along with either OBJTYPE(*LIB) or TYPE(04) SUBTYPE(01).

***MCHCTX:** The objects to be dumped are in the machine context. The following OS/400 system object types, whose MI system object names are given in parentheses, can reside *only* in the *machine* context: library (context), user profile, device (logical unit) description, network interface description, line (network) description, and control unit (controller) description. These types are included in the table given in the TYPE parameter description. *MCHCTX is valid only if one of these five object types is dumped.

context-name: Specify the name of the context containing the objects being dumped. The name of a library, such as QGPL or QTEMP, can be specified. If QTEMP is specified, the objects to be dumped are in the temporary job context associated with the job that this command is entered from or the job being serviced as a result of the Start Service Job (STRSRVJOB) command.

TYPE Specifies the type of MI objects to be dumped.

***ALL:** All MI object types in the specified context that have the specified name (if used) are dumped.

MI-object-type-in-hex: Specify the hexadecimal value that specifies the type of MI system objects to be dumped. The following table shows the MI system objects and their hexadecimal type codes. The value must be specified with both characters, but it does not have to be enclosed in apostrophes.

MI Type **MI System Object**	**Code**
Access group	01
Program	02
Module	03
Context (library)[1]	04
Byte string space	06
Journal space	07
User profile[1]	08
Journal port	09
Queue	0A
Data space	0B
Data space index	0C
Cursor	0D
Index	0E
Commit block	0F
Logical unit (device) description[1]	10
Network (line) description[1]	11
Controller (control unit) description[1]	12
Dump space	13
Class of Service space	14
Mode Description space	15
Network interface description[1]	16
Connection list	17
Queue space	18
Space	19
Process control space	1A
Authorization list space	1B
Dictionary space	1C
Machine context	81

MI Type	**Code**
MI System Object	

[1] If this object is specified for TYPE, then CONTEXT(*MCHCTX) must also be specified.

SUBTYPE

Specifies the subtype of the specified MI objects to be dumped, or specifies that all subtypes are being dumped.

***ALL:** All the subtypes of the specified MI objects are dumped.

MI-object-subtype-in-hex: Specify the specific subtype of the MI system objects to be dumped. Valid values range from 00 through FF. However, the subtype specified must be for an MI object actually in the specified context. If TYPE(*ALL) is specified, a specific subtype *cannot* be specified.

OBJTYPE

Specifies the object type of the OS/400 system objects to have their associated MI system objects dumped. If this parameter is specified, the TYPE or SUBTYPE parameters cannot be specified. More information on this parameter is in Commonly used parameters.

***ALL:** The specified MI objects of all OS/400 system object types are dumped.

object-type: Specify the specific OS/400 system object type whose associated MI system objects are to be dumped.

OFFSET

Specifies a list of values to use as offsets to indirectly address a single object that is being dumped. The values must be positive hexadecimal values or zeros that, when added to a pointer, result in valid addresses. If an offset of zero is added to a system pointer, the result is a space pointer to the start of the space associated with the object that is addressed by the system pointer. (In this discussion, the associated space of a space object is the space itself.)

Note: The OFFSET and SPACE parameters cannot be specified if *ALL or a generic object name is specified for the OBJ, TYPE, SUBTYPE, or OBJTYPE parameters.

***NONE:** No offset is specified. The object located through the context is dumped.

offset-value: Specify the list of offsets to pointers to use to address the object or space to be dumped. The values specified in this parameter are used as follows:

1. The first offset is added to a space pointer that points to the associated space of the object located through the context. The result is a space pointer that points to a location further into the space.
   a. If only one offset value is specified in this parameter, step 2 is not done and the dump operation, as indicated by the rest of the parameters in the command, is taken.
   b. If more than one offset is specified in this parameter, step 2 is repeated for each additional offset given.
2. Regarding the location pointed to by the space pointer produced in the previous step:
   a. If the location does not contain another pointer, the command is ended, an error message is sent to the user, and no dump operation is done.
   b. If the location contains a space pointer, the (next) offset is added to it. The result is another space pointer that points to either the same space or a different space or associated space.

c. If the location contains a system pointer, the associated space pointer is set from the system pointer, and the (next) offset is added to the space pointer. The result is a space pointer that points to a location in the associated space of the object addressed by the system pointer.

The result of step 2b or 2c is a space pointer that is used to perform step 2 again if there is another offset. If the last offset has been used, the final result is a location contained in a space pointer that is used as follows:

- If the resulting location contains a system pointer and the SPACE parameter is not specified, the system object pointed to by the system pointer is dumped. If the SPACE parameter is specified, the SPACE specification determines the portion of the system object that is dumped.
- If the resulting location contains a space pointer and the SPACE parameter is not specified, the portion of the space that starts at the location pointed to by the space pointer is dumped. If the SPACE parameter is specified, the SPACE specification determines the portion of the space to be dumped.

The following chart shows the offsets into the process control space (PCS) at which there are pointers to the components of a job structure. If one of these offsets is specified, OBJ(*PCS) must be specified or assumed.

Object **(Descriptive Name and Abbreviation)**	**Object Name**	**Offset**
Data management communications queue (DMCQ)	QDMCQ	20
Job temporary context (QTEMP)	QTEMP	40
MI response queue (MIRQ)	QMIRQ	80
Process definition template (PDT)	PDT	60
Process access group (PAG)	PAG	60 100
Spooling control block (SCB)	QSPSCB	200
Work control block table (WCBT)	QWCBT	10

SPACE

Specifies the area of a space or associated space to be dumped. The space is pointed to by the final pointer determined by the OFFSET parameter. If the OFFSET parameter is not specified, the final pointer is a system pointer to the specified object in the context. See *Note* in the OFFSET parameter description.

Element 1: Offset Value

*: If the final pointer is a system pointer, the object pointed to by that pointer is dumped. If the final pointer is a space pointer, the portion of the space that starts at the location pointed to by that pointer is dumped.

offset-value: Specify the value to add to the final pointer to point to the beginning of the area to dump. The value specified must be a positive hexadecimal value, ranging from 00000000 through 00FFFFFF, and, when added to the final pointer, results in a valid address.

Element 2: Length

*: The rest of the space pointed to as a result of the offset value is being dumped.

length: Specify a positive hexadecimal value ranging from 00000001 through 00FFFFFF that specifies the length of the area to be dumped. If the length specified is greater than the actual length of the space, only the actual space available is dumped.

Examples for DMPSYSOBJ

Example 1: Dumping Indexes

```
DMPSYSOBJ   CONTEXT(QTEMP)  TYPE(0E)
```

This command dumps the contents and attributes of all the indexes in the temporary job context to a spooled file for printing. MI indexes are identified by the type code 0E.

Example 2: Dumping a Device Description

```
DMPSYSOBJ   OBJ(WS1)  CONTEXT(*MCHCTX)  OBJTYPE(*DEVD)
```

This command dumps the device description for work station WS1, which is stored in the machine context.

Example 3: Dumping Process Control Space

```
DMPSYSOBJ   OBJ(*PCS)  SPACE(0  2A0)
```

This command dumps the work control block from the space associated with the process control space for the job.

Example 4: Specifying Offset Values

```
DMPSYSOBJ   OBJ(*PCS)  OFFSET(60 E0 10 10)  SPACE(0 20)
```

This command dumps the second call entry of the process automatic storage area (offset 60 E0) for a length of 32 bytes (SPACE(0 20)). If the third call level is dumped, OFFSET(60 E0 10 10 10) is specified.

Error messages for DMPSYSOBJ

***ESCAPE Messages**

CPF3502
No objects printed because no objects found.

CPF3508
SUBTYPE (&5) value is not permitted.

CPF3523
Starting offset &8 greater than size of space.

CPF3534
Not authorized to object.

CPF3537
Object &2 is damaged.

CPF3538
Cannot allocate object.

CPF3539
Object destroyed while being dumped.

CPF3560
Job being serviced not running.

CPF3561
Context &8 &9 &7 not found.

CPF3562
Object &7 not found.

CPF3563
Overflow value for file &1 in &2 too large.

CPF3566
No objects dumped because no objects found.

CPF3577
Data object &7 not found.

CPF3578
Base data object &7 not found.

CPF3642
Address of chain pointer &7 not permitted.

CPF3643
Address for chain pointer &7 not 16-byte aligned.

CPF3644
Base object &7 has no associated space.

CPF3645
Not authorized to base object &7.

CPF3646
Base object &2 is damaged.

CPF3647
Base object &8 or previous base object destroyed.

CPF3648
Base object &2 data area not found.

CPF3649
Chaining pointer &7 does not exist at location specified.

CPF3650
Chaining pointer &7 is instruction pointer.

CPF3651
Offset too large for base object &7.

CPF3652
Offset to last chaining pointer too large.

CPF3653
Location for last chaining pointer not 16-byte aligned.

CPF3654
Object &2 is damaged.

CPF3655
Last base object or final object previously deleted.

CPF3656
Base object &2 data area not found.

CPF3663
Base object number &7 not found.

CPF3664
Object &2 has no associated space.

CPF3665
Not authorized to dump object &2.

CPF3666
Object &2 is damaged.

CPF3667
Object to be dumped was destroyed.

CPF3668
Object &2 data area not found.

CPF3669
Final pointer does not exist at specified location.

CPF3670
Final pointer is instruction pointer.

CPF3671
Starting offset &8 too large.

CPF3672
Object specified by final pointer not found.

CPF3673
Not authorized to library &7.

CPF3909
Service command will not be processed.

CPF3913
Context &7 previously deleted.

CPF3914
Context &7 data area not found.

CPF3915
Context &7 damaged.

CPF3916
Context &7 not available.

CPF3918
Service request canceled.

CPF3925
Cannot open file &1.

CPF3935
Job being serviced ended during dump.

CPF3941
CONTEXT(*MCHCTX) and TYPE(&4) cannot be used together.

CPF3942
CONTEXT(*MCHCTX) and OBJTYPE(*&6) cannot be used together.

CPF3946
Context damaged.

CPF3947
Library &7 not available.

CPF3948
Library &3 previously deleted.

CPF3949
Library &7 damaged.

CPF3950
Error message &2 received for file &1. Request ended.

CPF3951
File &1 cannot be overridden by file name &2.

CPF3967
Dump not started because serviced job not running.

CPF3968
Dump not started because serviced job completed running.

CPF3969
Error during close of file &1. Output may not be complete.

DMPTAP (Dump Tape) Command Description

DMPTAP Command syntax diagram

Purpose

The DMPTAP (Dump Tape) command dumps label information, data blocks, or both from standard labeled tapes or tapes with no labels to a spooled printer file named QPTAPDMP. This command allows the user to dump one or more data files from the tape volume, writing the information to a printer file.

The tape volume being dumped must be on the specified device. After the DMPTAP command is entered, as much of the tape as necessary is read before the requested information is printed.

Data files on secured tapes can be dumped by the security officer only; any user can dump label information on secured tapes.

When the default values for the parameters of the DMPTAP command are used, the tape label areas and a minimal amount of data from the first file are printed. This command can help determine the record format of a data file on a tape with no label, or it can determine the exact contents of all label information for a labeled data file.

Required Parameter

DEV Specifies the name of the device in which the volume being dumped is placed. The volume may or may not be labeled. Specify the name of the tape or media library device.

Optional Parameters

VOL Specifies one or more volume identifiers used by the file. More information on this parameter is in commonly used parameters.

Note: If the device specified is a media library device, then the volume specified should be the cartridge identifier to be mounted and used.

***MOUNTED:** The volume on the specified device is dumped. The volume may or may not be labeled. Note that VOL(*MOUNTED) and LABEL(*NONE) must be specified to dump a volume that is not labeled. For a media library device, the volume to be used is the next cartridge in the category mounted by the Set Tape Category (SETTAPCGY) command.

volume-identifier: Specify the identifier of the labeled volume being dumped. This value can be specified only for dumping a labeled volume. If the tape on the specified device has a different volume identifier than the one specified in this parameter, or if it is not labeled, an error message is sent to the user of the DMPTAP command and the tape is not dumped.

SEQNBR
Specifies the range of sequence numbers for the data files that are dumped. If SEQNBR(*ALL) is specified, then all data files on the tape are dumped. If a range of sequence numbers is specified,

then only data files in that range of data file sequence numbers are dumped. The data files dumped may be further restricted by using the LABEL parameter.

The sequence number for a labeled tape data file is stored on labels ahead of and following the data in the file. For a volume that is not labeled, the data file sequence number is determined by the number of tape markers from the beginning of the tape. This parameter can be specified as a list of two values (Elements 1 and 2) or as a single value (*ALL or *SEARCH).

Element 1: First File to Dump

***FIRST:** The range of data files being dumped begins with the first file on the tape volume, regardless of its sequence number.

start-file-sequence-number: The range of data files being dumped begins with the data file with the specified sequence number. Specify a number that is less than or equal to the end-file-sequence-number value.

Element 2: Last File to Dump

***ONLY:** Only a single data file (specified by the start-file-sequence-number) is dumped.

***LAST:** The range of data files being dumped begins with the start-file-sequence-number data file and ends with the last data file on the end of the reel.

end-file-sequence-number: The range of data files being dumped ends with the specified sequence number data file. Specify a number that is greater than or equal to the start-file-sequence number.

Other Single Values

***ALL:** All data files on the volume on the specified device are dumped.

***SEARCH:** The volume that is on the device is searched for a data file with an identifier that matches the LABEL parameter value; when a match is found, the data file is dumped. If VOLLBL(*NO) is specified and the last operation on the device specified ENDOPT(*LEAVE) (the tape is positioned at the location where the last operation ended), the file search begins with the first data file beyond the current tape position. If VOLLBL(*YES) is specified or ENDOPT(*LEAVE) was not used for the last operation (or if the tape was manually rewound since an ENDOPT(*LEAVE) operation), the search begins with the first data file on the volume. SEQNBR(*SEARCH) is not valid when LABEL(*NONE) is specified, and cannot be used to dump a tape volume that is not labeled.

LABEL

Specifies the identifier of the specific data files that are dumped. The file identifier for a tape data file is stored on labels ahead of and following the data in the file.

***NONE:** All data files on the volume in the specified SEQNBR range are dumped. Note that VOL(*MOUNTED) and LABEL(*NONE) *must* be used to dump a tape volume that is not labeled.

file-identifier: Specify the data file identifier (17 alphanumeric characters maximum) of the data files being dumped. The system compares the LABEL identifier with the data file identifier on the labels of each file in the range specified by the SEQNBR parameter. All data files with an identifier that matches the LABEL identifier are dumped; any data file with an identifier that does not match the LABEL identifier is not dumped.

generic-file-identifier:* Specify a character string for a generic label identifier (17 alphanumeric characters maximum), which contains at least one character followed by an asterisk (*). Any tape file that has a file identifier with the same prefix as the generic data-file-identifier is dumped. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic names, refer to generic names.

TYPE Specifies the type of information that is being dumped. The dump may consist of the data file header labels or trailer labels (for a labeled tape volume only), data blocks from the data portion of the file, or all three types of information. If the tape volume on the device is not labeled, only *BASIC, *ALL, ≫ *HEX, ≪ or *DTABLK can be specified or an error message is sent to the user of the DMPTAP command and the volume is not dumped.

***BASIC:** For a standard-labeled volume, the dump includes header labels and the data blocks specified by the DTABLK parameter. For a volume that is not labeled, only the data blocks (DTABLK parameter) are dumped.

***ALL:** For a standard-labeled volume, the dump includes header labels, trailer labels, and data blocks. For a volume that is not labeled, TYPE(*ALL) dumps only data blocks (since there are no labels).

***NONE:** No data file is dumped. If TYPE(*NONE) is specified, the tape volume being dumped must be labeled, and VOLLBL(*NO) cannot be specified, or an error message is sent to the user of the DMPTAP command.

≫ ***HEX:** For a standard-labeled volume, the data is dumped as if the volume is a non-labeled tape. The header labels, data blocks, and trailer labels for a standard labeled file will appear to be three separate non-labeled tape files. For a non-labeled volume, the data is dumped the same as if *DTABLK is specified. ≪

***HDRLBL:** The data file header labels are dumped. Header labels are immediately ahead of the data in the file that they apply to. All header labels for the specified data files are dumped, including user-specified header labels (if any exist). TYPE(*HDRLBL) is not valid for volumes that are not labeled.

***DTABLK:** One or more data blocks from the file data are dumped. The blocks within the data file that are dumped are specified by the DTABLK parameter.

***TLRLBL:** All data file trailer labels are dumped. Trailer labels immediately follow the data in the file to which they apply. All the trailer labels for the specified data files are dumped, including user-specified trailer labels (if any exist). TYPE(*TLRLBL) is not valid for volumes that are not labeled.

DTABLK

Specifies which data blocks are dumped. This parameter is used to limit the amount of tape file data dumped to the printer. If neither TYPE(*BASIC) nor TYPE(*ALL) is specified and the TYPE parameter value does not include *DTABLK, this parameter is ignored.

This parameter can be specified as a list of two values (Elements 1 and 2) or as a single value (*ALL or *LAST).

Element 1: First Block to Dump

***FIRST:** The data blocks being dumped begins with the first block in the data file.

start-data-block: Specify the number of the first data block within each file that is dumped. If this number is greater than the number specified for the end-data-block portion of the DTABLK parameter, an error message is sent to the user who requested the dump, and the tape is not dumped. If the start-data-block value is larger than the actual number of data blocks in the data file, then the last data block in the file is dumped (with no error messages).

Element 2: Last Block to Dump

***ONLY:** Only the data block specified by the first part of the DTABLK parameter is dumped.

***LAST:** The range of data blocks that are dumped starts with the data block specified by the start-data-block value and goes to the last block in the file.

end-data-block: Specify the number of the last data block to dump within each file. If this number is less than the number specified for the start-data-block part of the DTABLK parameter, an error message is sent to the user who requested the dump, and the tape is not dumped. If the end-data-block value is larger than the actual number of data blocks in the data file, then all blocks from the start-block number to the end of the file are dumped (with no error messages).

Other Single Values

***ALL:** All data blocks in the specified data files on this volume are dumped. If a data file is continued from another volume or continues onto another volume, only the part of the data file that is stored on this volume is dumped.

***LAST:** Only the last data block in the data file is dumped.

VOLLBL

Specifies whether volume labels are dumped. This parameter is ignored for volumes that are not labeled.

***YES:** All volume labels (including user-specified labels) are dumped.

***NO:** No volume labels are dumped; the volume printout does, however, include the volume identifier of a labeled volume and other basic information for any dumped tape.

CODE Specifies the character code used. The code can be either extended binary-coded decimal interchange code (*EBCDIC) or the American National Standard Code for Information Interchange (*ASCII).

***EBCDIC:** The tape contains data in the EBCDIC character code. The dump output contains the hexadecimal value and the EBCDIC character equivalent of each data byte.

***ASCII:** The tape contains data in the ASCII character code. The dump output contains the hexadecimal value and the ASCII character equivalent of each data byte.

ENDOPT

Specifies whether the tape is rewound only or rewound and unloaded after the operation ends.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

Example for DMPTAP

```
DMPTAP  DEV(QTAPE2)  SEQNBR(5)  TYPE(*DTABLK)
  DTABLK(3 7)
```

This command dumps information from the tape volume that is on device QTAPE2. Data blocks 3 through 7 within the data file specified by sequence number 5 are dumped to a printer file.

Error messages for DMPTAP

***ESCAPE Messages**

CPF6708

Command ended due to error.

CPF6718

Cannot allocate device &1.

CPF6720

Incorrect volume &2 found on device &1.

CPF6721

Device &1 not a tape device.

CPF6723
File not found on volume &2 on device &1.

CPF6724
File label &5 not found on volume &2.

CPF6725
Ending file sequence number less than starting sequence number.

CPF6726
Ending data block less than starting block.

CPF6727
Dump type not allowed for nonlabeled volume on device &1.

CPF6728
LABEL(*NONE) CRTDATE(*NONE) required for nonlabeled volume.

CPF6729
No authority to file data on volume &2 device &1.

CPF6730
Cannot access file sequence number &5.

CPF6731
File label &5 not found on volume &2.

CPF6745
Device &1 not a media library device.

CPF6751
Load failure occurred on device &4.

CPF6760
Device &1 not ready.

CPF6772
Volume on device &1 cannot be processed.

CPF9814
Device &1 not found.

CPF9825
Not authorized to device &1.

CPF9845
Error occurred while opening file &1.

CPF9846
Error while processing file &1 in library &2.

CPF9847
Error occurred while closing file &1 in library &2.

CPF9850
Override of printer file &1 not allowed.

DMPTRC (Dump Trace) Command Description

DMPTRC Command syntax diagram

Purpose

The Dump Trace (DMPTRC) command copies data from the vertical microcode (VMC) trace table to a database file. The user can optionally run the job interactively or submit it as a batch job. Batch jobs will run under the submitter's job description and user profile.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. To use this command you must have *SERVICE special authority, or be authorized to the Service Trace function of Operating System/400 through iSeries Navigator's Application Administration support. The Change Function Usage Information (QSYCHFUI) API, with a function ID of QIBM_SERVICE_TRACE, can also be used to change the list of users that are allowed to perform trace operations.
3. The following user profiles have private authorities to use the command:
   - QSRV
   - QPGMR

Required Parameter

MBR Specifies the member name of the database file where the trace table data is dumped. If STRPFRMON was used to collect the trace data, the same name that was specified on the Start Performance Monitor (STRPFRMON) command should be used. If Performance Tools/400 is installed, this is a requirement for the reporting function of the Print Transaction Report (PRTTNSRPT) command.

Optional Parameters

LIB Specifies the library where the database file for trace data is located. If the file is not found in the specified library, the system automatically creates it in that library.

The name of the database file can be qualified by one of the following library values:

QPFRDATA: The data is located in the IBM-supplied performance data library, QPFRDATA.

library-name: Specify the name of the library to be searched.

JOBQ Specifies the qualified name of the job queue on which this job is placed. The name of the job queue can be qualified by one of the following library values:

QSYS: The IBM-supplied system library, QSYS, is used to locate the job queue.

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QCTL: The IBM-supplied controlling subsystem QCTL is used.

job-queue-name: Specify the name of the job queue.

***NONE:** No job is submitted. The DMPTRC request runs interactively.

TEXT Specifies the text that briefly describes the database member. More information on this parameter is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example for DMPTRC

```
DMPTRC  MBR(TUESAM)
```

This command causes existing VMC trace data to be written to the member TUESAM in library QPFRDATA. The file used is QAPMDMPT. The request is submitted to the job queue QCTL in library QSYS. It runs as a batch job.

Error messages for DMPTRC

***ESCAPE Messages**

CPF0A81
Performance trace cannot be sent to database file.

CPF0A82
Performance trace cannot be sent to database file.

CPF2110
Library &1 not found.

CPF3307
Job queue &1 in &2 not found.

CPF7207
Not able to start &2. Return code &3.

DMPUSRTRC (Dump User Trace Buffer) Command Description

DMPUSRTRC Command syntax diagram

Purpose

The Dump User Trace Buffer (DMPUSRTRC) command formats trace records in the user trace buffer for the specified job. Trace records are added to a user trace buffer by calling user trace facility APIs (Qp0zUprintf, Qp0zDump, Qp0zDumpStack, and Qp0zDumpTargetStack). The formatted trace records can be written to a database file or to the*stdout*special file.

Optional Parameters

JOB Specifies the job for which the user trace buffer is being dumped.

***:** The user trace buffer for the job that the command is running in is dumped.

job-name: Specify the name of the job whose user trace buffer is being dumped. If no user name or job number qualifier is given, all of the jobs currently in the system are searched for the simple job name. If duplicates of the specified name are found, a qualified job name must be specified.

user-name: Specify the name of the user of the job whose user trace buffer is being dumped.

job-number: Specify the six-digit number of the job whose user trace buffer is being dumped.

TRCRCDID
Specifies the record identifiers to be used in the formatted tracepoint records. Up to two identifiers can be specified.

***THD:** Thread identifiers are used. Each identifier contains eight hexadecimal digits.

***JOB:** Job identifiers are used. Each identifier contains the six-digit job number portion of the qualified job name.

OUTPUT
Specifies where the output from the command is directed to.

***FILE:** The output is written to database file QAP0ZDMP in library QTEMP.

***STDOUT:** The output is written to the*stdout*special file.

SLTTHD

Specifies a list of up to eight threads whose trace records are to be included. Only trace records for threads with the specified thread identifiers are included.

Note: This parameter and the OMTTHD parameter are mutually exclusive.

***ALL:** All trace records are included, unless excluded by another selection value.

thread-identifier: Specify the thread identifiers of up to eight threads whose trace records are to be included.

OMTTHD

Specifies a list of up to eight threads whose trace records are to be excluded. Trace records for all threads except those specified are included.

Note: This parameter and the SLTTHD parameter are mutually exclusive.

***NONE:** No trace records are excluded based on their thread identifier.

thread-identifier: Specify the thread identifiers of up to eight threads whose trace records are to be excluded.

Examples for DMPUSRTRC

Example 1: Dumping the Current User Trace Information

```
DMPUSRTRC
```

This command formats the user trace information for the current job and writes the formatted trace records to file QAP0ZDMP in library QTEMP.

Example 2: Dumping a Trace for a Specific Job

```
DMPUSRTRC JOB(004842/ACCT/WS6) OUTPUT(*STDOUT)
```

This command formats the user trace information for job WS6, which is associated with the user profile ACCT, and has the job number 004842, writing the formatted trace records to the*stdout*special file.

Error messages for DMPUSRTRC

***ESCAPE Messages**

CPFA98B

The User Trace buffer associated with job &3/&2/&1 could not be dumped.

CPFA98C

Job &3/&2/&1 not unique.

DUPDKT (Duplicate Diskette) Command Description

DUPDKT Command syntax diagram

Purpose

The Duplicate Diskette (DUPDKT) command copies the contents of a single diskette onto one or more diskettes. The volume table of contents (VTOC) and initial program load (IPL) record information and the data records can be copied to a diskette in a specified device. Diskette data in either the basic exchange data format or the save/restore E-format can be copied. The volume identifiers of the diskettes do not have to be unique.

If the diskettes to which data is being copied do not have the same sector size as the diskette from which data is being copied, then a message is sent to the system operator. The copying of diskettes may then be stopped, or the output diskette may be initialized to the same sector size as the input diskette before the copying continues.

Deleted sectors on the input diskette are ignored when copying. They are not copied to the new diskette. The address of the last data record in the file label containing the deleted sector is adjusted on the output diskette, according to the number of sectors found deleted. Therefore, if the input diskette has deleted sectors, the output diskette is not an exact copy.

Diskettes that have an extended label area (up to nine cylinders, in addition to cylinder 0, which are allocated as system area for data set labels), can be copied if the following conditions are followed:

- The RGZVOL option must be *NO.
- No deleted sectors can exist on the input diskette. If deleted sectors are found on the input diskette with an extended label area, a message is sent and the copying function is stopped.

Once a diskette has been copied, the Rename Diskette (RNMDKT) command can be used to rename the copied diskettes so that they have unique volume identifiers.

Restriction: A diskette cannot be copied if it contains control records that indicate records have been relocated, but not in sequence, or that sequential sector addresses are not in consecutive, ascending order. If deleted sectors are found on a diskette with an extended label area, the diskette cannot be copied.

In addition, the following restrictions apply to the type of diskette allowed by the Duplicate Diskette command:

1. A Type 1 diskette may only be copied to another Type 1 diskette.
2. A Type 2 diskette may be copied to a Type 2 diskette or Type 2D diskette. A Type 2D diskette is allowed because it was made for double-density recording. Thus, it is acceptable to use a Type 2D diskette for single-density recording. A message is sent to the system operator if an attempt is made to copy from a Type 2 diskette to a Type 2D diskette. The operator may then use the Initialize Diskette (INZDKT) command to initialize the Type 2D diskette to a Type 2 and continue processing. Note that the Type 2D now logically appears as a Type 2 diskette.
3. A Type 2D diskette may be copied to a Type 2 diskette or a Type 2D diskette. A Type 2 diskette is made for single-density recording and is more prone to media errors if used for double-density recording (Type 2D). If copying from a Type 2D diskette to a Type 2 diskette, a message is sent to the system operator. The operator may then cancel processing or use the Initialize Diskette (INZDKT) command to initialize the Type 2 diskette to a Type 2D diskette and continue processing. Note that the Type 2 diskette now logically appears as a Type 2D diskette and is more prone to media errors since it is meant for single-density recording.

Note: Results when copying to or from a diskette with labels that are not IBM standard labels are unpredictable. The diskette should be initialized by specifying CHECK(*NO) on the Initialize Diskette (INZDKT) command.

Required Parameters

FROMDEV
Specifies the name of the device from which the diskette is being copied.

TODEV
Specifies the name of the device to which the diskette is being copied.

Optional Parameters

RGZVOL

Specifies whether to delete the unused space between files to allow space for additional files to be written on the diskette.

***NO:** The unused space remains as it exists on the input diskette. If no deleted sectors are found, the output diskette is an exact copy of the input diskette.

***YES:** The unused space between files is moved to the end of the last file, making all files on the output diskette connected. If unused space between files currently exists, the files on the output diskette resides at different physical locations.

COPIES

Specifies, for spooled files, the number of copies being printed.

1: One copy of the output is printed.

number-of-copies: Specify the number of diskette copies to make. Valid values range from 2 through 999.

Examples for DUPDKT

Example 1: Copying Diskette Contents

```
DUPDKT  FROMDEV(DKT1)  TODEV(DKT2)
```

This command copies the entire contents of the diskette in device DKT1 onto the diskette in device DKT2.

Example 2: Copying Diskette Contents

```
DUPDKT  FROMDEV(DKT2)  TODEV(DKT2)
```

This command copies the entire contents of the diskette in device DKT2, prompts for the next diskette being inserted in device DKT2, and copies the contents on that diskette.

Example 3: Compressing Unused Space

```
DUPDKT  FROMDEV(DKT2)  TODEV(DKT1)  RGZVOL(*YES)
```

This command copies the contents of the diskette in device DKT2 to the diskette in device DKT1. The unused space of the data area is compressed at the end of the diskette. If unused space exists between files on the input diskette, the files on the output diskette reside at different physical locations.

Error messages for DUPDKT

*ESCAPE Messages

CPF5102

Permanent I/O error on volume &9 in device &4.

CPF6151

Cannot duplicate diskette in device &1.

CPF6157

Duplicate diskette ended; previous error occurred.

CPF6716

Device &1 not a diskette device.

CPF6718

Cannot allocate device &1.

CPF9814

Device &1 not found.

CPF9825

Not authorized to device &1.

DUPMEDBRM (Duplicate Media using BRM) Command Description

Note: To use this command, you must have the 5722-BR1 (Backup Recovery and Media Services for iSeries) licensed program installed. For detailed information on the parameters of this command, see the online help.

DUPMEDBRM Command syntax diagram

Purpose

The Duplicate Media using BRM (DUPMEDBRM) command copies the contents of a volume or volumes to another volume or set of volumes that you select.

Restrictions

- In a BRMS network, only volumes owned by the system doing the DUPMEDBRM can be duplicated. If a DUPMEDBRM command is used on a volume owned by another system, a BRM15A2 exception will be issued.
- You must have two devices to use this command. If the devices are shared non-MLB devices, BRMS will vary the devices on for you. If the devices are not shared devices, you must vary them on.
- The media or media set that you are copying must be members of the BRMS media inventory.
- If you are copying a media set in batch mode by specifying a volume in the media set in the VOL parameter, you must specify special value *SET in the FROMVOL parameter.

Notes:

1. The media devices do not have to support the same media classes. The media policy is specified in the DUPMEDBRM command.
2. The density field in the file header labels are updated to reflect the true density.
3. Do not precede an entry with an asterisk unless that entry is a “special value” that is shown (on the display itself or in the help information) with an asterisk.
4. If you are using DUPMEDBRM to copy a file group, the output media assumes the group number of the input media. An exception to this is if a SETMEDBRM command overrides the value for file group and type for the output volumes.
5. When you process the DUPMEDBRM command, media information is not automatically duplicated to the duplicate volume.
6. You cannot duplicate media for items saved to TSM servers using the DUPMEDBRM command.

Example for DUPMEDBRM

Example 1: Copying a Single Volume

```
DUPMEDBRM VOL(T00001) FROMDEV(TAP01)
TODEV(TAP03) FROMVOL(T00001)
```

In this example the volume T00001 is to be duplicated using TAP01 as the from device and TAP03 as the to device. Because T00001 is not a member of a media set, the input list (FROMVOL) contains only T00001.

Error messages for DUPMEDBRM

None

DUPOPT (Duplicate Optical) Command Description

DUPOPT Command syntax diagram

Purpose

The Duplicate Optical (DUPOPT) command creates a duplicate optical volume. The duplicate volume is identical to the original volume except for the volume identifier and time the volume was created.

Restriction: To use this command you must have *USE authority to the authorization list securing the source volume. You need *ALL authority to the authorization list securing the target volume if it is in an optical library device. You need *CHANGE authority to the authorization list securing the target volume if it is in an optical stand-alone device.

Required Parameters

FROMVOL

Specifies the volume identifier of the optical volume being duplicated. To determine the volume identifier of media not in an optical media library issue the following Display Optical CL command: DSPOPT VOL(*MOUNTED) DEV(device).

from-volume-identifier: Specify the source volume identifier.

***MOUNTED:** Use the optical volume mounted in the optical device specified by the FROMDEV parameter.

Note: This value is not valid for volumes in optical media library devices.

TOVOL

Specifies the volume identifier of the optical volume onto which the from-volume will be duplicated. This volume must have the following physical characteristics:

- If the FROMVOL media type is *WORM then the TOVOL media type can be either *WORM or *ERASE.
- If the FROMVOL media type is not *WORM then the TOVOL media type must be identical to the media type of the FROMVOL.
- If the TOVOL media type is *WORM it must be a volume that is not initialized.
- The TOVOL can not be on the opposite side of the FROMVOL cartridge.
- The block size of the FROMVOL and TOVOL must be identical.
- The volume capacity of the TOVOL must be equal to or greater than the FROMVOL volume capacity.

Use the Display Optical (DSPOPT) CL command to determine the physical characteristics of an optical volume.

to-volume-identifier: Specify the target volume identifier.

***MOUNTED:** Use the optical volume mounted in the optical device specified by the TODEV parameter.

Note: This value is not valid for volumes in optical media library devices.

Optional Parameters

NEWVOL
Specifies the new volume identifier of the to-volume after the duplication is complete.

***TOVOL:** The new volume identifier will be identical to the to-volume identifier.

new-volume-identifier: Specify the new volume identifier of the to-volume.

CLEAR
Indicates whether to re-initialize an optical volume if the volume is found to be already initialized.

Note: This parameter is ignored if the volume media type is *WORM.

***NO:** The volume is not re-initialized.

***YES:** The volume will be re-initialized.

Note: For media type *ERASE, specifying *YES will result in all existing data being erased prior to the start of the duplication process.

For media type *DVD-RAM, specifying *YES will not result in existing data being erased prior to the start of the duplication process. Though the data is not erased, access to the data is lost. If it is required that data on *DVD-RAM media be erased, initialize the volume using Initialize Optical (INZOPT), prior to running DUPOPT. Specify the CLEAR(*YES) parameter on the Initialize Optical (INZOPT) CL command.

FROMDEV
Specifies the name of an optical device which contains the from-volume.

Note: This parameter is only required if parameter FROMVOL is specified as *MOUNTED.

from-optical-device: The name of the optical device containing the from-volume.

TODEV
Specifies the name of an optical device which contains the to-volume.

Note: This parameter is only required if parameter TOVOL is specified as *MOUNTED.

to-optical-device: The name of the optical device containing the to-volume.

FROMENDOPT
After the DUPOPT request has completed, specifies whether to leave or unload the from-volume from the optical device in which it is located.

Note: This parameter is ignored if the from-volume is in an optical media library device.

***LEAVE:** The from-volume is left in the optical device.

***UNLOAD:** The from-volume is unloaded from the optical device.

TOENDOPT

After the DUPOPT request has completed, specifies whether to leave or unload the to-volume from the optical device in which it is located.

Note: This parameter is ignored if the to-volume is in an optical media library device.

***LEAVE:** The to-volume is left in the optical device.

***UNLOAD:** The to-volume is unloaded from the optical device.

Examples for DUPOPT

Example 1: Duplicate an optical volume when the volume names are known.

```
DUPOPT   FROMVOL(VOL01)  TOVOL(VOL02)
         NEWVOL(*TOVOL)  CLEAR(*YES)
```

This command creates a duplicate of the optical volume VOL01 on volume VOL02, which keeps the same volume identifier. VOL02 will be re-initialized prior to the duplication process.

Example 2: Duplicate an optical volume when the device names are known.

```
DUPOPT   FROMVOL(*MOUNTED)  TOVOL(*MOUNTED)
         NEWVOL(BKP001)     CLEAR(*YES)
         FROMDEV(OPT01)     TODEV(OPT02)
         FROMENDOPT(*LEAVE) TOENDOPT(*UNLOAD)
```

This command duplicates the optical volume in optical device OPT01 onto the volume in device OPT02. The optical volume in device OPT02 is re-initialized prior to the duplication process. The volume in device OPT01 will be left in the device after the duplication process completes. The volume in device OPT02 will be unloaded after the duplication process completes and will have a volume identifier of BKP001.

Error messages for DUPOPT

***ESCAPE Messages**

OPT1305
Optical volume &1 is read only.

OPT1315
Optical volume &1 is write protected.

OPT1320
Optical volume &1 in use.

OPT1325
Optical volume format not recognized.

OPT1330
Optical volume not found or not useable.

OPT1331
Optical volume &1 not found.

OPT1335
Volume &1 already initialized.

OPT1340
Optical volume &1 not initialized.

OPT1342
Invalid volume identifier specified.

OPT1346
Operation not allowed to volume located in a remote optical device.

OPT1350
Write operations failed to optical volume &1.

OPT1375
Optical volume &1 already exists.

OPT1460
Optical volume &1 is not in an optical device.

OPT1499
Source and target volumes are in different optical devices.

OPT1515
Unsupported or insufficient configuration on optical device &1.

OPT1530
&1 does not represent a valid optical device.

OPT1555
Optical device &1 in use.

OPT1605
Media or device error occurred.

OPT1790
Operation conflicts with another request.

OPT1805
Error accessing optical volume index file.

OPT1810
Error accessing optical directory index file.

OPT1815
Internal program error occurred.

OPT1820
Internal error occurred on optical device &1.

OPT1821
Internal error occurred on optical device &1.

OPT1825
Optical indexes are incorrect for optical device &1.

OPT1860
Request to optical device &1 failed.

OPT1861
No device description configured for resource &1.

OPT1862
No active device description for resource &1.

OPT1863
Optical libraries need to be reclaimed.

OPT1872
Optical request timed out.

OPT2029
TOVOL on opposite side of FROMVOL

OPT2050
The duplicate optical volume request from optical volume &1 to optical volume &2 failed.

OPT2301
Internal system object in use.

OPT2420
Not authorized to optical volume &2.

OPT7740
User not authorized to object &2 in library &3 type &4.

DUPTAP (Duplicate Tape) Command Description

DUPTAP Command syntax diagram

Purpose

The Duplicate Tape (DUPTAP) command copies the contents of one tape to another tape.

Notes:

1. The density field in the file header labels is updated to reflect the true density.
2. Byte 80 in the volume label of a tape written on device type 6157 is reset from a ’Q’ to a blank.

Restrictions:

1. The user must have two tape drives or a tape media library device with two tape resources to use this command.
2. A file that spans volumes must have both partial files duplicated at the same time. That is, duplicating a tape that ends in a partial file, followed by appending the second part of the file to the end of the tape is not allowed. You must duplicate both parts of the file at the same time by specifying multiple volumes on the FROMVOL parameter.

Required Parameters

FROMDEV
Specifies the name of the device from which the tape is copied.

TODEV
Specifies the name of the device to which the tape is copied.

Optional Parameters

FROMVOL
Specifies the volume identifier of the tape being duplicated, or indicates that the tape currently on the magnetic tape device is being duplicated.

Note: If the tape device is contained in a library device, then the volume specified should be the cartridge identifier to be mounted and used.

***MOUNTED:** The volume currently placed in the device is used.

volume-identifier: Specify the identifier of the labeled volume being duplicated. If the tape placed on the specified device has a different volume identifier than specified, or if it is an unlabeled volume, an error message is sent.

TOVOL

Specifies the volume identifiers of the tapes to be created (destination volumes).

Note: If the tape device is contained in a library device, then the volume specified should be the cartridge identifier to be mounted and used.

***MOUNTED:** The volume currently placed in the device is used. For a media library device, the volume to be used is the next cartridge in the category mounted by the Set Tape Category (SETTAPCGY) command.

***FROMVOL:** The volume label of the tape currently mounted on the source device is used to initialize the tape on the output device.

volume-identifier: Specify the identifier to use on the output volume.

FILES Specifies which data files are copied.

***ALL:** All data files on the tape volume are copied.

***ACTIVE:** Only data files with an expiration date later than the current system date are copied.

FROMSEQNBR

Specifies which data file sequence numbers are to be copied.

Element 1: Starting File Sequence Number

***FIRST:** All files starting with the first file sequence are duplicated.

***ALL:** All files are duplicated.

file-sequence-number: Specify the starting file sequence number to be duplicated. The valid range of sequence numbers is 1 through 16777215. Only the files in the specified sequence number range are duplicated.

Element 2: Ending File Sequence Number

***LAST:** All files ending with the last file sequence are duplicated.

***ONLY:** Only the file specified in the starting file sequence are duplicated. If *ALL is specified in the first element, then this parameter is ignored.

file-sequence-number: Specify the ending file sequence number of the range to be duplicated. The valid range of sequence numbers is 1 through 16777215.

TOSEQNBR

Specifies which sequence number the data files are to be copied to.

***FROMSEQ:** The data files are duplicated to the same file sequences as are specified in the from file sequence number parameter.

***END:** The data files are added to the logical end of tape. The next valid sequence number is used.

file-sequence-number: Specify the sequence number in which the data file will be copied to. This value is not allowed if the device does not have overwriting capabilities and the value specified is not the next logical value to be used at the end of the logical tape volume. » The valid range of sequence numbers is 1 through 16777215. « The duplication begins at the specified file.

TODENSITY

Specifies the density or format in which to write the data to the device specified on the TODEV parameter.

***DEVTYPE:** The highest capacity density or format supported by the tape device will be used.

Tape device	**Highest capacity density or format**
2440	6250
3422	6250
3430	6250
3480	*FMT3480
3490E	*FMT3490E
3570-BXX	*FMT3570
3570-CXX	*FMT3570E
3580-001	*ULTRIUM1
3590	*FMT3590
3590-Exx	*FMT3590E
6335	*QIC3040
6341	*QIC120
6342	*QIC525
6343	*QIC1000
6344	*QIC2GB
6346	*QIC120
6347	*QIC525
6348	*QIC1000
6349	*QIC2GB
6366	*QIC120
6368	*QIC1000
6369	*QIC2GB
6378	*QIC525
6379	*QIC1000
6380	*QIC2GB
6381	*QIC2DC
6382	*QIC4DC
6383	*QIC5010
6385	*QIC5010
6386	*MLR3
6387	*SLR100
6390	*FMT7GB

7207-122
*QIC4DC

7208-002
*FMT2GB

7208-012
*FMT5GB

7208-222
*FMT7GB

7208-342
*FMT20GB

9346 *QIC120

9347 3200

9348 6250

***CTGTYPE:** The highest capacity density or format supported by the device for the mounted cartridge type will be used. If the device does not support special cartridge type information, *DEVTYPE is used.

tape-density: Specify the density or format to use.

1600 The data density on the tape volume is 1,600 bits per inch, which is used for 1/2 inch reel tapes.

3200 The data density on the tape volume is 3,200 bits per inch, which is used for 1/2 inch reel tapes.

6250 The data density on the tape volume is 6,250 bits per inch, which is used for 1/2 inch reel tapes.

***FMT3480**
The format of this tape is FMT3480. The data density on this tape volume is formatted to support a 3480 device. This density is used for 1/2 inch cartridge tapes.

***FMT3490E**
The format of this tape is FMT3490E. The data density on this tape volume is formatted to support a 3490E device. This density is used for 1/2 inch cartridge tapes.

***FMT3570**
The format of this tape is FMT3570. The data format is written on the tape volume with a 3570 device.

***FMT3570E**
The format of this tape is FMT3570E. The data format is written on the tape volume with a 3570E device.

***FMT3590**
The format of this tape is FMT3590. The data format is written on the tape volume with a 3590 device. This density is used for 1/2 inch cartridge tapes.

***FMT3590E**
The format of this tape is FMT3590E. The data format is written on the tape volume with a 3590E device. This density is used for 1/2 inch cartridge tapes.

***QIC120**
The format of this tape is QIC120, which is used for 1/4 inch cartridge tapes that can hold 120 megabytes of data.

***QIC525**
The format of this tape is QIC525, which is used for 1/4 inch cartridge tapes that can hold 525 megabytes of data.

***QIC1000**
The format of this tape is QIC1000, which is used for 1/4 inch cartridge tapes that can hold 1200 megabytes of data.

***QIC2GB**
The format of this tape is QIC2GB. It is used by 1/4 inch tape devices which can store 2.5 gigabytes of data on a standard length QIC2GB cartridge.

***QIC2DC**
The format of this tape is QIC2DC. It is used to write compacted data to a 1/4 inch cartridge that supports the QIC2GB format.

***QIC4GB**
The format of this tape is QIC4GB. It is used by 1/4 inch tape devices which can store 4 gigabytes of data on a standard length QIC4GB cartridge.

***QIC4DC**
The format of this tape is QIC4DC. It is used to write compacted data to a 1/4 inch cartridge that supports the QIC4GB format.

***QIC3040**
The format of this tape is QIC3040, which is used for 1/4 inch minicartridge tapes that can hold 840 megabytes of data.

***QIC5010**
The format of this tape is QIC5010, which is used for 1/4 inch cartridge tapes that can hold 13.5 gigabytes of data.

***MLR3**
The format of this tape is MLR3. It is used by 1/4 inch tape devices which can store 25 gigabytes of data on a standard length MLR3 cartridge.

***SLR100**
The format of this tape is SLR100. It is used by 1/4 inch tape devices which can typically store 100 gigabytes of compacted data on a standard length SLR100 cartridge.

***FMT2GB**
The format of this tape is FMT2GB, which is used for 8 millimeter cartridge tapes that can hold 2 gigabytes of data.

***FMT5GB**
The format of this tape is FMT5GB, which is used for 8 millimeter cartridge tapes that can hold 5 gigabytes of data.

***FMT7GB**
The format of this tape is FMT7GB, which is used for 8 millimeter cartridge tapes that can hold 7 gigabytes of data.

***FMT20GB**
The format of this tape is FMT20GB. It is used by 8 millimeter tape devices that can store 20 gigabytes of data on a standard length cartridge.

***ULTRIUM1**
The format of this tape is ULTRIUM1. It is used by 1/2 inch cartridge tape devices that can store 100 gigabytes of data on a standard length cartridge.

Note: Some of the density values shown can only be specified when a tape device which supports that density is attached to the system.

Note: Self-configured tape devices may define additional valid values for the density parameter. Use iSeries 400 Operations Navigator (Configuration and Service) (Hardware) (Tape Units) (Properties) to find additional valid density values for a specific device, or use the F4=Prompt key on the Tape density field of the CL command to see a list of all valid density values for the attached tape devices.

COMPACT

Specifies whether device data compaction is performed. If the device specified does not support compaction, this parameter is ignored.

***FROMFILE:** Device data compaction is performed only if the file being read from the device specified on the FROMDEV parameter was written using device data compaction.

***YES:** Device data compaction is performed on all files written to the device specified on the TODEV parameter.

***NO:** Device data compaction is not performed.

USRLBLPGM

Specifies whether user tape labels are processed by a user program. The user label program passes user labels that are written to tape to the device specified on the TODEV parameter. The device specified on the FROMDEV parameter passes user labels to the user label program.

***SYSCOPY:** User tape labels are processed to allow proper duplication of System/36 save and restore tapes. If the tape volume specified on the FROMDEV parameter has user header labels, they are copied verbatim to the tape volume specified on the TODEV parameter. The same is done for the user trailer labels at the end of the file or for the trailer labels at the end of the file section.

If an end-of-volume condition occurs on the device specified on the TODEV parameter before the logical end-of-tape is found on the device specified on the FROMDEV parameter, user trailer and header labels are created and written to the current and next tape volumes on the device specified on the TODEV parameter. These labels replicate the data from the user header labels read at the beginning of the file.

***NONE:** No user tape labels are processed. Any user labels read from the tape volume are ignored. No user labels are written to the tape volume.

The name of the user label program can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

user-label-program-name: Specify the name of the user program that processes the user tape labels.

FROMENDOPT

Specifies whether the tape volume placed on the device specified on the FROMDEV parameter is rewound or rewound and unloaded after the operation is complete.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

TOENDOPT

Specifies whether the tape volume placed in the device specified on the TODEV parameter is rewound, or rewound and unloaded after the operation is complete.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

» **CHECK**

Specifies whether a tape file on the volume mounted on the **To device** is checked for active data before it is overwritten. If an unlabeled volume is on the **To device**, this parameter is ignored.

***YES:** The file to be overwritten is checked for active data. Only the first file to be overwritten is checked for active data, any subsequent files are not checked. If active files are found, the operation is ended and an error message is sent.

***NO:** Tape duplication continues with no checking for active files.

EXPDATE

Specifies the expiration date to be assigned to all the files when they are copied. This parameter only applies to standard labeled tapes.

***FROMFILE:** The expiration date currently specified for the file to be copied is used.

***PERM:** All the copied files will be assigned a permanent expiration date.

expiration-date: Specify the expiration date to be assigned to all the files when they are copied. «

Examples for DUPTAP

Example 1: Duplicating a single volume to a single volume

```
DUPTAP   FROMDEV(TAPE01)  TODEV(TAPE02)
```

This command duplicates the tape volume mounted on device TAPE01 onto the tape volume mounted on device TAPE02.

Example 2: Appending a volume set to the end of a single volume

```
DUPTAP   FROMDEV(TAPE01)  TODEV(TAPE02)
         FROMVOL( VOL001 VOL002)  TOVOL(VOLABC)
         FROMSEQNBR(*ALL) TOSEQNBR(*END)
```

This command duplicates all files from the tape volumes VOL001 and VOL002 onto the end of the to-volume VOLABC on device TAPE02.

Error messages for DUPTAP

*ESCAPE Messages

CPF67E8

FROMVOL and TOVOL parameters not correct

CPF67FA

Volume compatibility not correct.

CPF67FD
File sequence number &3 not correct for volume &2.

CPF67FE
No files found on volume &2.

CPF67F7
Continuation volume cannot be duplicated.

CPF67F8
TOSEQNBR not correct on volume &2.

CPF6703
Duplication to *END of partial file not correct.

CPF6704
Set not duplicated.

CPF6708
Command ended due to error.

CPF6718
Cannot allocate device &1.

CPF6720
Incorrect volume &2 found on device &1.

CPF6721
Device &1 not a tape device.

CPF6722
End of tape found on device &1.

CPF6734
File sequence number &3 not found on volume &2.

CPF6740
TODEV and FROMDEV must be different.

CPF6741
Nonlabeled tape format not valid on device &1.

CPF6745
Device &1 not a media library device.

CPF6751
Load failure occurred on device &4.

CPF6754
Active file &4 found on volume &2.

CPF6760
Device &1 not ready.

CPF6768
Volume on device &1 is write protected.

CPF6772
Volume on device &1 cannot be processed.

CPF9814
Device &1 not found.

CPF9825
Not authorized to device &1.

EDTAUTL (Edit Authorization List) Command Description

EDTAUTL Command syntax diagram

Purpose

The Edit Authorization List (EDTAUTL) command shows the list of users and their authorities. From this display, the user can add and remove users and change users' authorities on the authorization list.

Restriction: The user must have authorization list management authority to, or ownership of, the list to use this command.

Required Parameter

AUTL Specifies the name of an authorization list with which to work.

Example for EDTAUTL

```
EDTAUTL   AUTL(MYLIST)
```

This command shows the authorization list MYLIST and allows it to be changed.

Error messages for EDTAUTL

***ESCAPE Messages**

CPF22B9
Not authorized to change authorities.

CPF2204
User profile &1 not found.

CPF2207
Not authorized to use object &1 in library &3 type *&2.

CPF2208
Object &1 in library &3 type *&2 not found.

CPF2209
Library &1 not found.

CPF2211
Not able to allocate object &1 in &3 type *&2.

CPF2216
Not authorized to use library &1.

CPF2217
Not authorized to user profile &1.

CPF2283
Authorization list &1 does not exist.

CPF9843
Object &1 in library &3 type &2 cannot be accessed.

EDTBCKUPL (Edit Backup List) Command Description

EDTBCKUPL Command syntax diagram

Purpose

The Edit Backup List (EDTBCKUPL) command allows the user to select libraries and folders for backup. More information on backup is in the Backup, Recovery, and Availability topic in the Information Center.

Optional Parameter

BCKUPL

Specifies the backup list to be changed.

***LIB:** The library backup list is changed.

***FLR:** The folder backup list is changed.

Example for EDTBCKUPL

```
EDTBCKUPL  BCKUPL(*LIB)
```

This command displays the library backup list stored in user index QUSRSYS/QEZBACKUPL, and allows the user to change it.

Error messages for EDTBCKUPL

***ESCAPE Messages**

CPF1EEA

Not authorized to library backup list.

CPF1E6B

Folder backup list in use.

CPF1E6D

Folder backup list damaged; new one created.

CPF1E65

Library backup list in use.

CPF1E67

Backup options and library backup list damaged.

CPF1E99

Unexpected error occurred.

CPF7D41

Error occurred while logging order assistance request.

CPF7D42

Error occurred while performing database operation.

CPF9871

Error occurred while processing.

EDTCPCST (Edit Check Pending Constraints) Command Description

EDTCPCST Command syntax diagram

Purpose

The Edit Check Pending Constraints (EDTCPCST) command shows a list of established referential constraints that have records that are possibly in violation of the constraints (check pending). From this display, you can verify and select or change the sequence of the constraints to be rebuilt during an initial program load (IPL).

This command is called while you are running an attended IPL if you have check pending constraints. From the display shown, you can select whether the system continues the IPL while verifying selected constraints, or continues the IPL after verifying selected constraints.

There are no parameters for this command.

Example for EDTCPCST

Example 1: Editing a List of Constraints

```
EDTCPCST
```

This command shows you the referential constraints that are in check pending. You can edit the sequence for verifying the constraints from this display.

Error messages for EDTCPCST

***ESCAPE Messages**

CPF325C
Database object &1 is in error.

EDTIGCDCT (Edit DBCS Conversion Dictionary) Command Description

EDTIGCDCT Command syntax diagram

Purpose

The Edit DBCS Conversion Dictionary (EDTIGCDCT) command lets the user add, change, and delete alphameric entries and their related words from the specified double-byte character set (DBCS) conversion dictionary. The system refers to the DBCS conversion dictionary when doing DBCS conversion. The system displays the entries being edited when this command is specified.

Note: Use of the conversion function is not recommended for Chinese and Korean DBCSs.

Required Parameter

IGCDCT
Specifies the qualified name of the DBCS conversion dictionary being edited. If a library name is not specified, the first dictionary found when searching the library list is edited.

The name of the dictionary can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

dictionary-name: Specify the name of the dictionary that will be edited.

Optional Parameter

ENTRY

Specifies the alphanumeric entries being edited with their related words.

***ALL:** Any entry in the dictionary can be edited. The system first shows the Work with DBCS Conversion Dictionary display showing all alphanumeric entries in the dictionary. From this display, specific entries are chosen to be edited.

generic-string:* Specify the generic name of the string to be edited. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic names, refer to generic names.

specific-string: Specify the entry to be edited. The system displays the Edit Related Words display, showing a single alphanumeric entry and its related words. You can edit the related words on this display. The string cannot be longer than 12 characters.

Examples for EDTIGCDCT

Example 1: Showing the Work with DBCS Conversion Dictionary Display

```
EDTIGCDCT   IGCDCT(DBCSLIB/QUSRIGCDCT)  ENTRY(123*)
```

This command shows the Work with DBCS Conversion Dictionary display showing all the alphanumeric entries that start with 123 in the dictionary QUSRIGCDCT, which is stored in the library DBCSLIB.

Example 2: Showing the Edit Related Words Display

```
EDTIGCDCT   IGCDCT(DBCSLIB/QUSRIGCDCT)  ENTRY(WORDS)
```

This command shows the Edit Related Words display showing the alphanumeric entry WORDS and its related words from the dictionary QUSRIGCDCT, which is stored in library DBCSLIB.

Additional Considerations

Consider the following when using this command:

- You can only rearrange the words related to alphanumeric entries when editing the IBM-supplied DBCS conversion dictionary (QSYS/QSYSIGCDCT). You cannot add or remove related words or alphanumeric entries.
- Editing can only be done at a display station. The following display stations are needed:
  – If a specific string is specified with the ENTRY parameter or to display DBCS characters, use a DBCS-capable display station.
  – If a specific string is not specified with the ENTRY parameter, or if the user does not want to display DBCS characters, use either a DBCS-capable display station or a 24-row by 80-column alphanumeric display station.
- How to specify specific and generic strings for the ENTRY parameter:
  – You must enclose a string of characters in apostrophes (’) if the string to be edited contains Katakana, lowercase alphabetic, or special symbols. The system translates lowercase alphabetic characters to uppercase characters unless they are specified as parameter values in apostrophes. For example, specify the string abc as ’abc’.
  – When specifying a string that contains only numeric or uppercase alphabetic characters, the string does not have to be enclosed in apostrophes (’). That means that ABC is as acceptable to the system as ’ABC’.
  – Do not include embedded blanks ( ) in a string of characters. In other words, the system would not accept the string ABC D.

- Specify the correct library name.
- The first display produced by the EDTIGCDCT command, the Work with Japanese Dictionary display, shows all of the dictionary entries. The second display, the Edit Related Words display, shows individual entries and their related words.

Error messages for EDTIGCDCT

***ESCAPE Messages**

CPF2122
Storage limit exceeded for user profile &1.

CPF8138
&8 damage on DBCS conversion dictionary &4 in &9.

CPF8440
Entries cannot be added to the system DBCS conversion dictionary.

CPF8451
Entry value &1 not correct.

CPF8455
Work station is not a DBCS device.

CPF8461
Entry &1 of DBCS conversion dictionary is logically damaged.

CPF9801
Object &2 in library &3 not found.

CPF9802
Not authorized to object &2 in &3.

CPF9803
Cannot allocate object &2 in library &3.

CPF9810
Library &1 not found.

CPF9820
Not authorized to use library &1.

CPF9845
Error occurred while opening file &1.

CPF9846
Error while processing file &1 in library &2.

CPF9847
Error occurred while closing file &1 in library &2.

EDTDOC (Edit Document) Command Description

EDTDOC Command syntax diagram

Purpose

The Edit Document (EDTDOC) command allows the user to edit a document using the word processing function of OfficeVision. More information on editing documents is in the Using OfficeVision/400 Word Processing book.

Optional Parameters

DOC Specifies the name of the document to edit.

***PRV:** The name used in the previous session is used.

document-name: Specify the name of the document that is edited.

FLR Specifies the name of the folder that contains the document.

***PRV:** The name used in the previous session is used.

folder-name: Specify the name of the folder that contains the document to be edited.

EXITPNL
Specifies whether the Exit Document display is shown when F3(Exit) or F12(Cancel) is pressed to end the editing.

***YES:** The Exit Document display is shown when F3(Exit) or F12(Cancel) is pressed to end the editing.

***NO:** The Exit Document display is not shown when F3(Exit) or F12(Cancel) is pressed to end the editing.

Example for EDTDOC

```
EDTDOC   DOC(TASK4) FLR(INSTTXT)
```

This command displays the document TASK4 of the folder INSTTXT, and allows the user to edit the document TASK4.

Error messages for EDTDOC

***ESCAPE Messages**

OFCFFFC
User storage capacity exceeded.

OFCFFFD
Damaged object found.

OFC8EA3
OfficeVision for AS/400 editor is not available to resolve to a display.

OFC80B5
OfficeVision for OS/400 editor is not available on the system.

OFC800A
Folder is in use.

OFC800B
Document &1 is in use.

OFC800F
Display does not support text.

OFC8006
Folder not found.

OFC8007
Document &1 not found in folder.

OFC8008
Request not allowed with folder.

OFC8009
Request not allowed with document &1.

OFC801A
Document has been saved to diskette, tape or save file.

OFC801D
Maximum number of text sessions active.

OFC801E
DW editor or text assist cannot be loaded.

OFC8010
Document &1 cannot be processed.

OFC8011
Document &1 needs to be recovered.

OFC8016
Document &1 is checked out.

OFC8018
Document &1 is empty.

OFC8019
Required module not on system.

OFC802E
Request failed for PC editor.

OFC821B
Document &1 needs to be reclaimed.

OFC8951
Data name must be specified.

OFC8952
Type must be &9 or &10.

OFC8953
Data &9 does not exist.

OFC8954
Display terminal does not have graphics ability.

OFC8955
PC Text-assist function required to view image.

OFC903A
Document &1 is final form.

OFC9811
Folder needs to be reclaimed.

EDTDLOAUT (Edit Document Library Object Authority) Command Description

EDTDLOAUT Command syntax diagram

Purpose

The Edit Document Library Object Authority (EDTDLOAUT) command is used to change authorization to a document and/or folder object.

The following information is shown for the specified document or folder:

- The name of the document library object

- The name of the authorization list securing the document or folder (if there is one)
- Personal status of the document library object
- Specific user authority for the document or folder
- The authority given to the users with no specific authority for the document or folder
- Access codes can be shown by pressing a function key

Restrictions:

1. You must have *ALL authority to the document or folder to change the authority or *ALLOBJ special authority.
2. You must have authority to use the ADDDLOAUT, CHGDLOAUT, and the RMVDLOAUT commands to use this command.
3. You must have *ALLOBJ special authority to change the *ROOT folder public authority.

Required Parameter

DLO Specifies the name of the document or folder that is changed.

***SYSOBJNAM:** The system object name specified in the SYSOBJNAM parameter is changed.

***ROOT:** The public authority value of the *ROOT folder is changed.

document-library-object-name: Specify the user-assigned name of the document or folder that is changed. Up to 12 characters can be specified.

Optional Parameters

FLR Specifies the name of the folder that contains the document.

***NONE:** A folder name is not specified. If DLO(name) is specified and the object is located in a folder, FLR(*NONE) cannot be specified.

folder-name: Specify the name of the folder that contains the object. The name can consist of a series of folder names if the folder containing the object is located in another folder. Up to 63 characters can be specified.

SYSOBJNAM
Specifies the system object name of the folder or document. This parameter is valid only when DLO(*SYSOBJNAM) is specified. Ten characters must be specified.

Example for EDTDLOAUT

```
EDTDLOAUT  DLO(DOCA)  FLR(MYFLR)
```

This command allows the user of this command to change the list of authorized users and their authorities to the document library object named DOCA in folder MYFLR. The user of this command must have *ALL authority to the object or be the owner of the object.

Error messages for EDTDLOAUT

***ESCAPE Messages**

CPF8A78
Folder &1 in use.

CPF8A79
Folder &1 is logically damaged.

CPF8A80
Document &2 in use in folder &1.

CPF8A82
Document &2 not found in folder &1.

CPF8A88
Operation not allowed on document &2 in folder &1.

CPF8A89
Document &2 in folder &1 is logically damaged.

CPF89C0
You have specified an incorrect value.

CPF90BA
Authority request for document library object failed.

CPF90B6
You have specified an incorrect input value.

CPF9073
No authority to view or change the security of document library object &1.

CPF9079
Request to get document description not successful for user profile &1.

CPF908A
Requester &1 not enrolled.

CPF908B
Document library object not found.

CPF909A
Document &2 in folder &1 is damaged.

CPF9095
Folder &1 is damaged.

CPF9845
Error occurred while opening file &1.

CPF9846
Error while processing file &1 in library &2.

CPF9847
Error occurred while closing file &1 in library &2.

EDTF (Edit File) Command Description

EDTF Command syntax diagram

Purpose

The Edit File (EDTF) command allows you to edit a stream file or a database file member.

Restriction: For database files, the record length of the file cannot exceed 4096.

Required Parameters

STMF Specifies the name of the file to be edited. If the file specified is a directory instead of a file, a list of objects in the directory is shown. Either this parameter or the FILE parameter is required.

’stream-file-path-name’: Specify the path name of the stream file or a pattern to match the name of the stream file to be edited.

The path name can be either a simple name or a name that is qualified with the name of the directory in which the file is located. A pattern can be specified in the last part of the path name. An asterisk (*) matches any number of characters. If the path name is qualified or contains a pattern, it must be enclosed in apostrophes.

FILE Specifies the name of the database file that contains the member to be edited. If no library qualifier is specified, the library list is used to locate the file. Either this parameter or the STMF parameter is required.

The name of the database file member can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file that contains the member to be edited.

Optional Parameter

MBR Specifies the name of the database file member to be edited.

***FIRST:** The first member in the database file is edited.

member-name: Specify the name of the file member to be edited.

Examples for EDTF

Example 1: Editing a Stream File

```
EDTF  STMF('/mydir/myfile.txt')
```

This command will start an edit session for file myfile.txt in directory mydir under the root directory.

Example 2: Editing a Database File Member

```
EDTF  FILE(MYLIB/MYFILE) MBR(MYMBR1)
```

This command will start an edit session for member MYMBR1 of file MYFILE in library MYLIB.

No error messages.

EDTLIBL (Edit Library List) Command Description

EDTLIBL Command syntax diagram

Purpose

The Edit Library List (EDTLIBL) command shows an entry display that allows users to make changes to the library list. This command cannot be used to change the system portion of the library list or the library list of any other job.

Restriction: This command is valid only in an interactive environment.

There are no parameters for this command.

Example for EDTLIBL

```
EDTLIBL
```

This command shows the Edit Library List display from which the user can add libraries, remove libraries, and change the order of the libraries in the library list.

Error messages for EDTLIBL

***ESCAPE Messages**

CPF2106
Library list not changed.

CPF2184
Library list not replaced.

EDTOBJAUT (Edit Object Authority) Command Description

EDTOBJAUT Command syntax diagram

Purpose

The Edit Object Authority (EDTOBJAUT) command displays the list of authorized users of an object and their associated authorities. If you own the object or have *ALLOBJ special authority, you can add, change, or remove authority for the object. If you have object management authority for the object, you can remove your specific authorities or grant or remove them for other users.

The following are displayed for the specified object:

- the object name
- the name of the library containing the object
- the name of the object's owner, the object's type, and a list of all the users who are authorized to use the object
- the authorities that each user has for the object
- the authorization list name if the object is secured by an authorization list.

If an object does not have an owner name associated with it, no authorities for the object are shown.

Restrictions:

1. The user must have object management authority to the object to use this command.
2. If the object is a file, the user must have object operational and object management authorities.
3. » You must have *USE authority to the auxiliary storage pool device if one is specified.«

Required Parameters

OBJ Specifies the qualified name of the object for which the authorized users and their authorities are to be shown.

The name of the object can be qualified by one of the following library values:

<u>*LIBL:</u> All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the name of the object for which its authorized users and their authorities are to be shown.

OBJTYPE

Specifies the object type of the object that is to have its authorized users and their authorities shown. Any one of the operating system object types can be specified. For example, to display a list of users authorized to use a file, specify the value *FILE. More information on this parameter is in Commonly used parameters.

» Optional Parameter

ASPDEV

Specifies the auxiliary storage pool (ASP) device name where the library that contains the object (OBJ parameter) is located. If the object's library resides in an ASP that is not part of the library name space associated with the job, this parameter must be specified to ensure the correct object is used as the target of the edit authority operation.

***:** The ASPs that are currently part of the job's library name space will be searched to locate the object. This includes the system ASP (ASP number 1), all defined basic user ASPs (ASP numbers 2-32), and, if the job has an ASP group, all independent ASPs in the ASP group.

***SYSBAS:** The system ASP and all basic user ASPs will be searched to locate the object. No independent ASPs will be searched, even if the job has an ASP group.

auxiliary-storage-pool-device-name: The device name of the independent ASP to be searched to locate the object. The independent ASP must have been activated (by varying on the ASP device) and have a status of 'Available'. The system ASP and basic user ASPs will not be searched. «

Example for EDTOBJAUT

```
EDTOBJAUT  OBJ(ARLIB/PROG1)  OBJTYPE(*PGM)
```

This command causes the list of authorized users and their authorities for the object named PROG1 to be shown, but only if the user has object management authority for the object. PROG1 is a program (*PGM) located in the library named ARLIB.

Error messages for EDTOBJAUT

*ESCAPE Messages

CPF22B8

Not authorized to change authorities.

CPF22B9

Not authorized to change authorities.

CPF2204

User profile &1 not found.

CPF2207

Not authorized to use object &1 in library &3 type *&2.

CPF2208

Object &1 in library &3 type *&2 not found.

CPF2209
Library &1 not found.

CPF2211
Not able to allocate object &1 in &3 type *&2.

CPF2216
Not authorized to use library &1.

CPF2217
Not authorized to user profile &1.

CPF2283
Authorization list &1 does not exist.

» **CPF980B**
Object &1 in library &2 not available.

CPF9814
Device &1 not found.

CPF9825
Not authorized to device &1. «

CPF9843
Object &1 in library &3 type &2 cannot be accessed.

» **CPF9873**
ASP status is preventing access to object. «

EDTQST (Edit Questions and Answers) Command Description

EDTQST Command syntax diagram

Purpose

The Edit Questions and Answers (EDTQST) command allows authorized users to edit questions and answers for publication in a specified database. More information is available in the Basic System Operations topic in the Information Center.

Restrictions:

1. A user must have authority to the command and be a Q & A coordinator for any Q & A database referred to by the command.

Optional Parameters

QSTDB
Specifies the Questions-and-Answers (Q & A) database in which to edit questions and answers.

***SELECT:** The user is asked to specify a Q & A* database. If only one Q & A database exists on the system, it is the default.

question-database: Specify the name of the Q & A database in which to edit questions and answers.

LIB
Specifies the name of the library that contains the Q & A database.

***QSTLIB:** The library containing the specified Q & A* database is searched. If *SELECT is specified on the QSTDB parameter, any Q & A database in any library for which the user is authorized can be selected.

library-name: Specify the name of the library to be searched. If *SELECT is specified on the QSTDB parameter, any database in the library for which the user is authorized can be selected.

Example for EDTQST

```
EDTQST
```

This command shows the Work with Candidate Questions display.

Error messages for EDTQST

None

EDTRBDAP (Edit Rebuild of Access Paths) Command Description

EDTRBDAP Command syntax diagram

Purpose

The Edit Rebuild of Access Paths (EDTRBDAP) command shows the editing options that are available to the user when rebuilding an access paths. From this display, users can selectively control the rebuilding of access paths.

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QSYSOPR user profile has private authority to use the command.
2. If an access path is marked with HELD, the user cannot open the access path or run a query that uses the access path.

Example for EDTRBDAP

```
EDTRBDAP
```

This command shows the controls that are available when editing rebuild access paths.

Error messages for EDTRBDAP

***ESCAPE Messages**

CPF325C
Database object &1 is in error.

EDTRCYAP (Edit Recovery for Access Paths) Command Description

EDTRCYAP Command syntax diagram

Purpose

The Edit Recovery for Access Paths (EDTRCYAP) command shows a list of access path recovery times for the system and for auxiliary storage pools (ASP) that are currently active on the system. From this list, you can change target access path recovery times and view updated recovery status information. » Additionally, the command will show up to 500 access paths with the largest estimated access path recovery time which are not eligible for system-managed access-path protection and why they are not eligible. Also, the command will show up to 500 access paths with the largest estimated access path recovery time which are currently being protected by system-managed access-path protection. «

The system uses no more than the specified amount of target access path recovery time when recovering access paths during an initial program load (IPL) after an abnormal system end. Because access path recovery time is a target, performance may range around the target.

The time taken to rebuild access paths exposed while running the Copy File (CPYF), the Reorganize Physical File Member (RGZPFM), or the Restore Object (RSTOBJ) commands is not considered in the target access path recovery time of access paths protected with this command.

You can use this command or the Change Recovery for Access Paths (CHGRCYAP) command to manage the protection of access paths that are not already protected through journaling.

For more information on using this command, see the Backup, Recovery, and Availability topic in the Information Center.

This command has no parameters.

Restrictions:

1. You must have job control special authority to use this command.
2. This command is shipped with public *EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to use this command.
3. If the current access path recovery state is *OFF, the user must be in a restricted state to activate system-managed access-path protection by specifying a target access path recovery time value.
4. If no user auxiliary storage pools (ASPs) exist on the system, an access path recovery time for ASP 1 cannot be specified. You must specify a system access path recovery time.

Example for EDTRCYAP

```
EDTRCYAP
```

This command shows the Edit Recovery for Access Paths display from which you can show or modify the target access path recovery times for your system and configured ASPs.

Error messages for EDTRCYAP

<u>*ESCAPE Messages</u>

CPF70FA
Recovery times reset before changes completed.

CPF70FB
No authority to use command.

CPF70FC
ASP time changes not valid with system time of *OFF.

CPF70FE
ASP time changes not valid when system time is *OFF.

CPF70F4
Error occurred.

CPF70F7
Restricted system required to change recovery times.

CPF70F9
Not all recovery time changes made active.

CPF700F
Access path recovery time for &1 set to *NONE.

CPF701C
Change to system access path recovery time canceled.

CPF701D
Error occurred during change of recovery times.

CPF701E
Access path protection cannot be turned *OFF.

CPF702E
Access path recovery times set to system defaults.

CPF9871
Error occurred while processing.

EDTS36PRCA (Edit System/36 Procedure Attributes) Command Description

EDTS36PRCA Command syntax diagram

Purpose

The Edit System/36 Procedure Attributes (EDTS36PRCA) command presents the attributes of the specified procedure on your display for you to change. The attributes of a specified procedure or of all procedures in the specified file can be changed.

Required Parameter

MBR Specifies the name of the procedure member for which attributes are shown or updated.

***ALL:** The attributes of all procedure members in the file are shown for update.

procedure-member-name: Specify the name of the procedure member whose attributes you want to update.

Optional Parameter

FILE Specifies the qualified name of the source physical file that contains the procedure members to be updated.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QS36PRC: The name of the default source physical file that contains the procedure members is used.

source-file-name: Specify the name of the source physical file that contains the procedure members.

Example for EDTS36PRCA

```
EDTS36PRCA   MBR(RPGPROC) FILE(RPGLIB)
```

This command shows the attributes of procedure RPGPROC in file QS36PRC in library RPGLIB and allows them to be changed.

Error messages for EDTS36PRCA

***ESCAPE Messages**

CPF2C0A
Member &3 attributes not changed.

CPF2C0B
Changing attributes not allowed for SSP member &3.

CPF2C08
File &1 is not a source file.

CPF7D41
Error occurred while logging order assistance request.

CPF7D42
Error occurred while performing database operation.

CPF9803
Cannot allocate object &2 in library &3.

CPF9812
File &1 in library &2 not found.

CPF9815
Member &5 file &2 in library &3 not found.

CPF9820
Not authorized to use library &1.

CPF9822
Not authorized to file &1 in library &2.

CPF9826
Cannot allocate file &2.

CPF9871
Error occurred while processing.

EDTS36PGMA (Edit System/36 Program Attributes) Command Description

EDTS36PGMA Command syntax diagram

Purpose

The Edit System/36 Program Attributes (EDTS36PGMA) command presents the attributes of the specified program on your display to allow you to change them. The attributes of a specified program or of all programs in the specified library can be changed.

Required Parameter

PGM Specifies the qualified name of the program having its attributes updated.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***ALL:** The attributes of all programs in the library are shown for update. *ALL is not allowed if the library specified is *LIBL.

program-name: Specify the name of the program whose attributes you want to update.

Example for EDTS36PGMA

```
EDTS36PGMA   PGM(RPGLIB/*ALL)
```

This command shows the program attributes of all the programs in RPGLIB and allows them to be changed.

Error messages for EDTS36PGMA

***ESCAPE Messages**

CPF2C01
Program &1 attributes not changed.

CPF2C02
Changing attributes not allowed for SSP program &1.

CPF2C03
MRTMAX parameter value &3 not correct.

CPF2C05
Program name *ALL not allowed with library *LIBL.

CPF7D41
Error occurred while logging order assistance request.

CPF7D42
Error occurred while performing database operation.

CPF9803
Cannot allocate object &2 in library &3.

CPF9811
Program &1 in library &2 not found.

CPF9820
Not authorized to use library &1.

CPF9830
Cannot assign library &1.

CPF9871
Error occurred while processing.

EDTS36SRCA (Edit System/36 Source Attributes) Command Description

EDTS36SRCA Command syntax diagram

Purpose

The Edit System/36 Source Attributes (EDTS36SRCA) command presents the attributes of the specified source member on your display for you to change. The attributes of a specified source member or of all source members in the specified file can be changed.

Required Parameter

MBR Specifies the name of the source member that is having its attributes shown or updated.

***ALL:** The attributes of all source members in the file are shown for update.

source-member-name: Specify the name of the source member whose attributes you want to update.

Optional Parameter

FILE Specifies the qualified name of the source physical file containing the source members.

The name of the source physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QS36SRC: The name of the default source physical file containing the source members.

source-file-name: Specify the name of the source physical file containing the source members.

Example for EDTS36SRCA

```
EDTS36SRCA   MBR(*ALL) FILE(SDALIB/QS36SRC)
```

This command shows the source attributes of all the source members in file QS36SRC in library SDALIB and allows them to be changed.

Error messages for EDTS36SRCA

***ESCAPE Messages**

CPF2C0A
Member &3 attributes not changed.

CPF2C0B
Changing attributes not allowed for SSP member &3.

CPF2C08
File &1 is not a source file.

CPF7D41
Error occurred while logging order assistance request.

CPF7D42
Error occurred while performing database operation.

CPF9803
Cannot allocate object &2 in library &3.

CPF9812
File &1 in library &2 not found.

CPF9815
Member &5 file &2 in library &3 not found.

CPF9820
Not authorized to use library &1.

CPF9822
Not authorized to file &1 in library &2.

CPF9826
Cannot allocate file &2.

CPF9871
Error occurred while processing.

EDTWSOAUT (Edit Workstation Object Authority) Command Description

EDTWSOAUT Command syntax diagram

Purpose

The Edit Workstation Object Authority (EDTWSOAUT) command shows the list of authorized users of a workstation object and the users' associated authorities. Workstation objects are used by the OS/400 Graphical Operations program. If you own the object or are the security officer, you can add, change, or remove authority for the object. If you have object management authority for the object, you can remove your specific authorities or grant or remove them for other users.

The following are shown for the specified object:
- The object name
- The name of the library containing the object
- The name of the object owner, the object's type, and a list of all the users who are authorized to use the object
- The authorities that each user has for the object

If an object does not have an owner name associated with it, no authorities for the object are shown.

Restrictions:
1. The user must have object management authority to the object to use this command.
2. If the object is a file, the user must have object operational and object management authorities to use this command.

Required Parameter

WSOTYPE

Specifies the name of the workstation object for which specific authorities are to be shown or edited.

The special values for this parameter are described in the following table.

Special Value	Workstation Objects
*TPLWRKARA	Work area template
*WRKARA	Work area objects
*TPLPRTOL	Printer output list template
*PRTOL	Printer output list objects
*TPLTPRTL	Printer list template
*PRTL	Printer list objects
*TPLOUTQ	Output queue template
*TPLOUTQL	Output queue list template
*OUTQL	Output queue list objects
*TPLJOBL	Job list template
*JOBL	Job list objects
*TPLJOBQ	Job queue template
*TPLJOBLOG	Job log template
*JOBLOG	Job log objects
*TPLJOBQL	Job queue list template
*JOBQL	Job queue list objects
*TPLMSGL	Message list template
*MSGL	Message list objects
*TPLMSGQ	Message queue template
*TPLMSGSND	Message sender template
*MSGSND	Message sender
*TPLSGNUSL	Signed-on user list template
*SGNUSL	Signed-on user list objects
*TPLOBJL	Object list template
*OBJL	Object list objects
*TPLLIBSL	Library list template
*LIBSL	Library list objects
*TPLLIB	Library template
*TPLLAUNCH	Job submitter template
*LAUNCH	Job submitter objects
*PRSSET	Personal setting objects

Example for EDTWSOAUT

```
EDTWSOAUT   WSOTYPE(*TPLMSGQ)
```

This command shows the list of authorized users to the message queue template.

No error messages.

EJTEMLOUT (Eject Emulation Output) Command Description

EJTEMLOUT Command syntax diagram

Purpose

The Eject Emulation Output (EJTEMLOUT) command finishes the current emulation output, and then ejects to a new page. It forces the last data received from the host system to the spooled file or printer by closing the printer file and then reopening another, so that more data can be spooled or printed. If SCHEDULE(*FILEEND) is specified on the Create Printer File (CPTPRTF) command, printing starts. The request may not take effect immediately, because the printer utility must complete printing a transmission block from the host system before doing this function. See “Additional Considerations” at the end of this command description for more information about running this command.

More information about device emulation is available in the 3270 Device Emulation Support book.

Required Parameters

EMLDEV
Specifies the name of the printer emulation device requested to receive data from the host system. The printer emulation job using this device is informed of the request and closes the printer file. This forces all of the data received from the host system to the spooled file or printer. The printer file is then reopened and printer emulation continues. To use this function, the user must be authorized to the device.

EMLLOC
Specifies the remote location name associated with this session. This name is defined during configuration and refers to the remote location where communication takes place. This value must be the same as that specified for the Start Printer Emulation (STRPRTEML) command.

PRTDEV
Specifies the name of the printer device that is used to print the spooled output. This value must match the value specified on the Start Printer Emulation (STRPRTEML) command. This parameter must be specified when the EMLLOC parameter is specified.

Example for EJTEMLOUT

```
EJTEMLOUT   EMLDEV(HOSTPRT1)
```

This command closes the printer file in the printer emulation job using the emulation device HOSTPRT1, forcing the latest data from the host system out to the spooled file or printer.

Additional Considerations

You must use care when running this command. Before entering the command, you should look at the printed output (if SPOOL(*NO) was specified) or use the Display Spooled File (DSPSPLF) command to look at the spooled file (if SPOOL(*YES) was specified), to determine whether the printer data is at a logical breaking point. If this function is requested when printer emulation is in the middle of a group of print data from the host system, the group is split into separate printer files on the system.

The effect of this command on the printer emulation output varies, depending on the values specified for the SPOOL and SCHEDULE parameters on the printer file.

The possible values and their conditions are:

- **SPOOL(*NO):** All the data received from the host system is printed, and the printer moves to the top of the next page.
- **SPOOL(*YES) and SCHEDULE(*IMMED):** If a writer is active to the output queue and is printing this file, all the data received from the host system is printed, and the printer moves to the top of the next page. If a writer is not active (printing this file), the effect is the same as if SCHEDULE(*FILEEND) was specified. Another printer file is opened on the output queue.
- **SPOOL(*YES) and SCHEDULE(*FILEEND):** The status of the printer file on the output queue changes from *open* to *ready to print*. If a writer is active, the data can be printed. Another printer file is opened on the output queue.
- **SPOOL(*YES) and SCHEDULE(*JOBEND):** The status of the printer file on the output queue changes from *open* to *closed*. The file is not ready to print until the end of the job is reached. Another printer file is opened on the output queue.

Error messages for EJTEMLOUT

***ESCAPE Messages**

CPF8595
Eject emulation output function not performed.

ELSE (Else) Command Description

ELSE Command syntax diagram

Purpose

The Else (ELSE) command is used with an IF command to specify another command that is to be conditionally processed. The ELSE command is processed only if the result of evaluating the logical expression on the preceding IF command is false, the ELSE command can specify a CL command to be processed for the false condition. If the DO command is specified on the ELSE command, a Do group can be processed. If the result of processing the IF command is true, the ELSE command and commands associated with it are not processed.

An ELSE command does not have to follow each IF command, but each ELSE command that is coded must have an associated IF command preceding it. If nested levels of IF commands are used, a given ELSE is always matched with the innermost IF command that has not already been matched with another ELSE command. Although the ELSE command is optional, coding all of the matching ELSE commands makes it easier to see where all of the nesting levels start and end.

Restriction: The ELSE command is valid only in a CL program. It must have an associated IF command preceding it.

Optional Parameter

CMD Specifies the command or commands (in a Do group) to be processed if the result of evaluating the expression on the corresponding IF command is false. If the command specified in this parameter is a DO command, all of the commands specified within the Do group are considered to be part of the command specified by the parameter. If no command is specified, no action is taken for a false condition.

If the command specified by the CMD keyword is not coded on the same line as the keyword, the left parenthesis following CMD must be coded on the same line, followed by a + or - to show continuation. The command and the right parenthesis can then be coded on the next line. For example:

```
ELSE  CMD(  +
    GOTO C)
```

If any part of the command continues on the next line, a continuation character (+ or -) must be specified.

If a DO command is specified, only the DO command (not the commands specified as part of the Do group) is placed in parentheses. For example:

```
ELSE  CMD(DO)
    CMD1
    CMD2
      *
      *
      *
    ENDDO
```

The following commands, although valid in CL programs, cannot be specified on the ELSE command:

ENDDO (End Do)

MONMSG (Monitor Message)

PGM (Program)

ENDPGM (End Program)

DCL (Declare CL Variable)

DCLF (Declare File)

another ELSE command

In addition, the MONMSG command cannot be specified as the next command after the ELSE command.

Examples for ELSE

Example 1: Using Else and If Commands

```
IF   (&A *GT &B)  THEN(CHGVAR  VAR(&A)  VALUE(&B))
  ELSE   (CHGVAR &B &A)
```

If the value of &A is greater than the value of &B, &A is set equal to &B. If &A is less than or equal to &B, the test result is false. The CHGVAR command on the ELSE command is processed, and the value of &B is set to the same value as &A. (Refer to the CHGVAR (Change Variable) command for the description of the command and its parameters.)

Example 2: Nested Levels of Commands

```
IF   COND(&A *EQ &B)  +
  THEN(IF   (&C *EQ &D)  +
    THEN(IF   (&C *EQ &F)   THEN(DO)))
                        CMD1
                        CMD2
                         *
                         *
                         *
                        ENDDO
          ELSE   CMDX
     ELSE   CMDY
ELSE   DO
```

This example shows the use of nested levels of IF commands where an ELSE command is associated with each IF. The use of the ELSE commands makes the nested levels of IF commands easier to identify.

Error messages for ELSE

***ESCAPE Messages**

CPF2150
Object information function failed.

CPF2151
Operation failed for &2 in &1 type *&3.

CPF8122
&8 damage on library &4.

CPF8123
Damage on object information for library &4.

CPF8129
Program &4 in &9 damaged.

CPF9803
Cannot allocate object &2 in library &3.

CPF9806
Cannot perform function for object &2 in library &3.

CPF9807
One or more libraries in library list deleted.

CPF9808
Cannot allocate one or more libraries on library list.

CPF9810
Library &1 not found.

CPF9811
Program &1 in library &2 not found.

CPF9820
Not authorized to use library &1.

CPF9821
Not authorized to program &1 in library &2.

CPF9830
Cannot assign library &1.

CPF9871
Error occurred while processing.

EMLPRTKEY (Emulate Printer Key) Command Description

EMLPRTKEY Command syntax diagram

Purpose

The Emulate Printer Key (EMLPRTKEY) command causes the printer emulation job or session that is using the specified printer emulation device to send either a PA1 or PA2 key signal to the host system. PA keys are program attention keys that are used to signal the host system. The command may not take effect immediately because the printer emulation routine will complete printing the last data received from the host system before doing this function. In addition, the PA key signal, although sent to the host system, may not immediately be received.

Required Parameters

EMLDEV
Specifies the name of a printer emulation device that receives data from the host system. This

device must be a 3287 Printer (EMLDEV(3287)) or a 3289 Printer (EMLDEV(3289)), and must currently be operating as an LU1 unit. The printer emulation job or session that is using this device will be informed of the request. If the LU1 session is between brackets, printer emulation starts a bracket and sends the PA key signal to the host system with Change Direction (CD) request. If the LU session is in receive condition, a signal (request for CD) is sent to the host system, and printer emulation waits for the CD. When the CD is received, the PA key signal is sent to the host system with the CD. If the LU session is in send condition, the PA key signal is sent to the host system with the CD.

EMLLOC

Specifies the remote location name associated with this session. This name is defined during configuration and refers to the remote location where communication takes place. This value is the same as the value specified for the EMLLOC parameter for the Start Printer Emulation (STRPRTEML) command.

PRTDEV

Specifies the name of the printer device that is used to print the spooled output. This value must match the value specified on the Start Printer Emulation (STRPRTEML) command. This parameter must be specified when the EMLLOC parameter is specified.

PRTKEY

Specifies the PA key signal that is sent to the host system. The host system program determines how the PA keys work.

***PA1:** The PA1 key signal is sent to the host system.

***PA2:** The PA2 key signal is sent to the host system.

Example for EMLPRTKEY

```
EMLPRTKEY  EMLDEV(HOSTPRT2)  PRTKEY(*PA2)
```

This command causes the printer emulation session using emulation device HOSTPRT2 to send the PA2 key signal to the host system.

Error messages for EMLPRTKEY

***ESCAPE Messages**

CPF8598

Emulate print key function not performed.

ENDASPBAL (End ASP Balance) Command Description

ENDASPBAL Command syntax diagram

Purpose

The End ASP Balance (ENDASPBAL) command allows the user to end the auxiliary storage pool (ASP) balance function that was started using the Start ASP Balance (STRASPBAL) CL command. A message will be sent to the system history (QHST) log when the ASP balance function is ended for each ASP.

For more information about ASP balancing see the Hierarchical Storage Management book.

Restrictions: You must have *ALLOBJ special authority to use this command.

» **ASP**

Specifies the auxiliary storage pool (ASP) number for which the ASP balancing function is to be ended. A value must be specified for the ASP parameter or the ASPDEV parameter.

***ALL:** ASP balancing will be ended for the system ASP (ASP number 1) and all basic ASPs (ASP numbers 2-32) defined to the system.

auxiliary-storage-pool-number: Specify the ASP for which ASP balancing is to be ended. Valid ASP numbers are 1 to 32. Up to 32 ASP numbers may be specified.

ASPDEV

Specifies the name of the auxiliary storage pool (ASP) device for which the ASP balancing is to be ended. A value must be specified for the ASP parameter or the ASPDEV parameter.

***ALLAVL:** ASP balancing will be ended for all ASP devices that currently have a status of ’Available’.

auxiliary-storage-device-name: Specify the name of the independent ASP device for which ASP balancing is to be ended. Up to 32 ASP device names may be specified.

Examples for ENDASPBAL

Example 1: End ASP Balance for ASP 1

```
ENDASPBAL  ASP(1)
```

This command allows the user to end the ASP balancing function for ASP 1.

Example 2: End ASP Balance for ASPs 1-32

```
ENDASPBAL  ASP(*ALL)
```

This command will end the ASP balancing functions for the system ASP (ASP number 1) and each basic ASP (ASP numbers 2-32) that is currently being balanced.

Example 3: End ASP Balance for an ASP Device

```
ENDASPBAL  ASPDEV(MYASP1)
```

This command will end the ASP balancing function for ASP device MYASP1.

Example 4: End ASP Balancing for All ASPs

```
ENDASPBAL  ASP(*ALL) ASPDEV(*ALLAVL)
```

This command will end the ASP balancing functions that may be active on ASP numbers 1-32 and all ASP devices that have a status of ’Available’.

Error messages for ENDASPBAL

*ESCAPE Messages

CPF18AC

ASP balancing not active for ASP &2.

CPF18AD

ASP &2 must contain more than a single unit.

CPF1890

*ALLOBJ authority required for requested operation.

CPF9829

Auxiliary storage pool &2 not found.

ENDBCHJOB (End Batch Job) Command Description

ENDBCHJOB Command syntax diagram

Purpose

The End Batch Job (//ENDBCHJOB) command is a delimiter that indicates the end of a job in a batch job stream. The //ENDBCHJOB command also can indicate the end of an inline data file if the command is detected while the inline file is being processed. Other conditions that can indicate the end of a job are:

- Batch Job (BCHJOB) command
- An end-of-file condition while an input stream is being processed

Restriction: The ENDBCHJOB command cannot be used from a work station. Two slashes must precede this command name when entering it in the data record, for example, `//ENDBCHJOB`. The user can separate the slashes from this command name with blank spaces, for example, `//  ENDBCHJOB`.

There are no parameters for this command.

Example for ENDBCHJOB

```
//ENDBCHJOB
```

This command indicates the end of a job that began with the BCHJOB command.

Error messages for ENDBCHJOB

***ESCAPE Messages**

CPF1753
Command cannot be run.

ENDCLNUP (End Cleanup) Command Description

ENDCLNUP Command syntax diagram

Purpose

The End Cleanup (ENDCLNUP) command allows the user to end the cleanup operation. Any active batch cleanup jobs, either processing or on the job queue, are ended immediately.

This command does not alter the parameters specified on the Change Cleanup (CHGCLNUP) command. The cleanup operation can be restarted by specifying the Start Cleanup (STRCLNUP) command.

More information on cleanup commands is in the Basic System Operations topic in the Information Center.

Restriction: The user must have *JOBCTL authority to use this command.

There are no parameters for this command.

Example for ENDCLNUP

```
ENDCLNUP
```

This command ends the cleanup operation.

Error messages for ENDCLNUP

*ESCAPE Messages

CPF1E2A
Unexpected error in QSYSSCD job.

CPF1E2B
Power scheduler and cleanup options not found.

CPF1E33
Cleanup options or power schedule in use by another user.

CPF1E35
Not authorized to end cleanup.

CPF1E36
Cleanup has not been started.

CPF1E99
Unexpected error occurred.

»

ENDCLUNOD (End Cluster Node) Command Description

ENDCLUNOD Command syntax diagram

Purpose

The End Cluster Node (ENDCLUNOD) command is used to end Cluster Resource Services on one or all the nodes in the membership list of an existing cluster. The status of each node that is ended is set to Inactive. In order to restart Cluster Resource Services on nodes that have been ended, the Start Cluster Node (STRCLUNOD) command is used.

When a node in the cluster is ended, it is not removed from the cluster membership list.

This command can be called on the node which is to be ended, or it can be called on any node in the cluster which has a status of Active. If this command is called when the cluster is partitioned, only nodes in the partition running the command will process the request.

The cluster resource group exit program on the node being ended will be called with an action code of End Node. The exit program on all other nodes in the recovery domain will be called with an action code of Failover. If all the nodes in the cluster are being ended, cluster resource group exit programs will not be called with an indication to failover.

The recovery domain of cluster resource groups on the node that had ended will indicate a node status of Active even though the node is inactive. For all the other nodes in the recovery domain, the status of the node will be Inactive. If the node being ended is the primary node for an active device cluster resource group, ownership of the hardware associated with the cluster resource group will be moved to a backup node. If the cluster resource group is not active, there are no backup nodes, or all backup nodes are either inactive or in a different cluster partition, the ownership of the hardware is left with the node being ended.

Restrictions

1. To use this command you must have *IOSYSCFG authority.
2. This CL command cannot be called from a cluster resource group exit program.
3. The node being ended must be active.

Required Parameters

CLUSTER
Specifies the name of the cluster that contains the node or nodes being ended.

cluster-name: Specify the name of the cluster.

NODE Specifies the node identifier(s) being ended.

***ALL:** End all active nodes in the cluster.

node-identifier: Specify the name of the node being ended.

Optional Parameter

OPTION
Specifies the method to end the node:

***IMMED:** Immediate. The request to end Cluster Resource Services on the node will be processed immediately.

***CNTRLD:** Controlled. Pending cluster resource group actions will complete before the request to end Cluster Resource Services is processed.

Example for ENDCLUNOD

```
ENDCLUNOD CLUSTER(MYCLUSTER) NODE(NODE01) OPTION(*IMMED)
```

This command ends Cluster Resource Services on node NODE01 for cluster MYCLUSTER. The request is processed immediately without waiting for any pending cluster resource group actions to complete.

Error messages for ENDCLUNOD

***ESCAPE Messages**

CPF0001
Error found on &1 command.

«»

ENDCRG (End Cluster Resource Group) Command Description

ENDCRG Command syntax diagram

Purpose

The End Cluster Resource Group (ENDCRG) command disables resiliency of the specified cluster resource group. The cluster resource group status is set to Inactive. The resources associated with the cluster resource group are no longer resilient. That is, there is no failover or switchover action provided for these resources while the cluster resource group is ended.

Ending a device cluster resource group will not change the ownership of devices. The devices remain on whatever nodes owns them at the time the command is run. Also, the devices are not varied off when the cluster resource group is ended.

If an exit program is specified for the cluster resource group, it is called with an action code of End on each active node in the recovery domain. When the exit program is called, the cluster resource group status is set to End Cluster Resource Group Pending. Successful completion of the exit program sets the cluster resource group status to Inactive. In addition, for an application cluster resource group:

- the current application exit program job on the primary node will be cancelled with the *IMMED option
- the takeover IP interface for the cluster resource group will be ended.

If the exit program fails and the original state of the cluster resource group cannot be recovered, the cluster resource group status is set to Indoubt.

Restrictions

1. To use this command you must have *IOSYSCFG authority.
2. This command cannot be called from a cluster resource group exit program.
3. Cluster Resource Services must be started on the node processing the request.
4. The status of the cluster resource group being ended must be Active or Indoubt.

Required Parameters

CLUSTER

Specifies the name of the cluster containing the cluster resource group.

cluster-name: Specify the name of the cluster.

CRG

Specifies the name of the cluster resource group which will be ended.

cluster-resource-group-name: Specify the name of the cluster resource group to end.

Optional Parameter

EXITPGMDTA

Specifies up to 256 bytes of data that is passed to the cluster resource group exit program when it is called. This parameter may contain any scalar data except pointers. For example, it can be used to provide state information. This data will be stored with the specified cluster resource group and copied to all nodes in the recovery domain. Pointers in this area will not resolve correctly on all nodes and should not be placed in the data. The data specified will replace the existing exit program data stored with the cluster resource group. This parameter must be set to *SAME if no exit program is specified for the cluster resource group.

<u>*SAME:</u> The exit program data stored with the cluster resource group specified will be passed to the exit program.

exit-program-data: Specify the data that is to be passed to the exit program.

Example for ENDCRG

```
ENDCRG CLUSTER(MYCLUSTER) CRG(MYCRG) EXITPGMDTA('important information')
```

This command ends resiliency of the cluster resource group called MYCRG in the cluster called MYCLUSTER. When the cluster resource group exit program is called, it will be passed the exit program data 'important information' on all active nodes in the recovery domain.

Error messages for ENDCRG

<u>*ESCAPE Messages</u>

CPF0001

Error found on &1 command.

«

ENDCHTSVR (End Clustered Hash Table Server) Command Description

ENDCHTSVR Command syntax diagram »

Purpose

The End Clustered Hash Table Server (ENDCHTSVR) command is used to end the specified clustered hash table server on the cluster nodes specified by the NODE parameter. This will remove the specified nodes from the clustered hash table domain. If all of the cluster nodes defined in the clustered hash table domain are specified in the NODE parameter the server job will be ended on all cluster nodes and the clustered hash table server will be deleted from the cluster. The clustered hash table server was started using the Start Clustered Hash Table Server (STRCHTSVR) command. If the clustered hash table server has any active connections on the nodes being removed from the domain, any future requests from those connections will fail.

Restrictions

- Cluster Resource Services must be active on the local node.
- All nodes specified in the NODE parameter must have Cluster Resource Services active.
- If an authorization list was specified when the server was started the requesting user must have *CHANGE authority to the authorization list.

Required Parameter

SERVER
Specifies the name of the clustered hash table server to be ended.

server-name: The name of the clustered hash table server to be ended.

Optional Parameter

NODE Specifies which nodes will end the clustered hash table server. The nodes specified will be removed from the clustered hash table domain. If all the nodes in the clustered hash table domain are specified (or *ALL), the server will no longer exist in the cluster. Nodes in this list must be unique. The nodes must be active in the cluster.

<u>*LOCAL:</u> The clustered hash table server will be ended on the local node only. *LOCAL can be specified only once in the list of nodes specified.

***ALL:** The clustered hash table server will be ended on the all nodes in the clustered hash table domain. If specified, *ALL must be the only value in the list.

node-identifier: Specify the nodes to process the end request. Up to 128 cluster nodes can be specified.

Examples for ENDCHTSVR

Example 1: Ending a clustered hash table server on the local node

```
ENDCHTSVR SERVER(CT0)
```

This command ends the clustered hash table server CT0 on the local node only.

Example 2: Ending a clustered hash table server on one of two nodes in the clustered hash table domain

Domain for clustered hash table CT1 is FRED and BARNEY.

```
ENDCHTSVR  SERVER(CT1) NODE(FRED)
```

This command ends the clustered hash table server CT1 on cluster node FRED. The clustered hash table server is still active on BARNEY.

Example 3: Ending a clustered hash table server on all nodes

Domain for clustered hash table CT2 is FRED and BARNEY.

```
ENDCHTSVR  SERVER(CT2) NODE(*ALL)
```

This command ends the clustered hash table server named CT2 on the local node (i.e. BARNEY) and node FRED. The clustered hash table will not exist in the cluster after this command runs.

Error messages for ENDCHTSVR

***ESCAPE Messages**

CPFBD03
End Clustered Hash Table Server failed with reason code &1.

«

ENDCMTCTL (End Commitment Control) Command Description

ENDCMTCTL Command syntax diagram

Purpose

The End Commitment Control (ENDCMTCTL) command ends the commitment definition associated with the activation group for the program that issued the command. Changes to commitment resources associated with the commitment definition are no longer made after this command is processed.

This command either ends the activation group level or the job level commitment definition associated with the activation group for the program that issued the command. A commitment definition is first established by the Start Commitment Control (STRCMTCTL) command.

If there are uncommitted changes for an interactive job, a message is sent asking the user whether the changes should be committed or rolled back before a commitment definition is ended. For a batch job, the changes are rolled back.

More information on the use of commitment control is in the Commitment control article in the Information Center.

There are no parameters for this command.

Example for ENDCMTCTL

```
ENDCMTCTL
```

This command specifies that the commitment definition established with the STRCMTCTL command is to end. The system determines if any changes have been made to the commitment resources after the last commitment boundary (at the last completed Commit (COMMIT) command or Rollback (ROLLBACK) command). If changes have been made for an interactive job, a message is sent asking the user whether the changes should be made permanent (committed) or removed (rolled back). For batch jobs, any changes are rolled back.

Error messages for ENDCMTCTL

***ESCAPE Messages**

CPF83E4
Commitment control ended with resources not committed.

CPF835A
End of commitment definition &1 canceled.

CPF835B
Errors occurred while ending commitment control.

CPF835C
Commitment control ended with remote changes not committed.

CPF8350
Commitment definition not found.

CPF8355
ENDCMTCTL not allowed. Pending changes active.

CPF8356
Commitment control ended with &1 local changes not committed.

CPF8367
Cannot perform commitment control operation.

ENDCMNSVR (End Communications Server) Command Description

ENDCMNSVR Command syntax diagram

Purpose

The End Communications Server (ENDCMNSVR) command is used to end the target display station pass-through server. The target display station pass-through server processes iSeries 400 display station pass-through, iSeries 400 Client Access work station function (WSF), and other 5250 emulation programs on programmable workstations.

Restrictions:

You must have job control (*JOBCTL) special authority to use this command.

Optional Parameters

OPTION
Specifies whether the target display station pass-through server is ended in an immediate or controlled manner.

***CNTRLD:** The server is ended in a controlled manner. Active sessions are allowed to complete their processing. New sessions are not allowed. After the specified period of time elapses, the processing for ENDCMNSVR OPTION(*IMMED) is performed.

***IMMED:** The server is ended in an immediate fashion. All active sessions that were started through the target display station pass-through server are ended immediately.

DELAY
Specifies the amount of time (in seconds) allowed in which to complete a controlled end of the target display station pass-through server. After this period of time all the target display station pass-through server jobs are ended immediately.

***NOMAX:** There is no maximum amount of time to wait. The servers will not end until all active sessions end normally.

delay-time: Specify the number of seconds in which the end operation is completed. Valid values range from 1 through 86400 seconds.

Example for ENDCMNSVR

Example 1: Ending Target Display Station Pass-through Server

```
ENDCMNSVR
```

This command ends the target display station pass-through server in a controlled manner. Any active sessions that are using the target display station pass-through server are not affected. New sessions are not allowed through the target display station pass-through server. Once all of the active sessions have ended, the target display station pass-through server will end.

Error messages for ENDCMNSVR

ENDCMNTRC (End Communications Trace) Command Description

ENDCMNTRC Command syntax diagram

Purpose

The End Communications Trace (ENDCMNTRC) command ends the trace running on the specified line, network interface, or network server description.

Restrictions:

1. The user must have *USE authority for the line, network interface, or network server to be traced.
2. To use this command you must have *SERVICE special authority, or be authorized to the Service Trace function of Operating System/400 through iSeries Navigator's Application Administration support. The Change Function Usage Information (QSYCHFUI) API, with a function ID of QIBM_SERVICE_TRACE, can also be used to change the list of users that are allowed to perform trace operations.
3. This command is shipped with public *EXCLUDE authority.
4. The following user profiles have authority to this command:
   - QSECOFR
   - QSRV

Required Parameters

CFGOBJ
Specifies the name of the configuration object being traced. The object is either a line description, a network interface description, or a network server description.

CFGTYPE
Specifies the type of configuration description being traced.

***LIN:** The type of configuration object is a line description.

***NWI:** The type of configuration object is a network interface description.

***NWs:** The type of configuration object is a network server description.

Example for ENDCMNTRC

```
ENDCMNTRC  CFGOBJ(*QESLINE)  CFGTYPE(*LIN)
```

This command ends the communications trace of line description QESLINE.

Error messages for ENDCMNTRC

***ESCAPE Messages**

CPF26AE
Network server description &1 not found.

CPF2601
Line description &1 not found.

CPF2634
Not authorized to object &1.

CPF39AE
Trace already ended.

CPF39AF
Trace is ending - please wait

CPF39A7
Trace storage not available in communications processor

CPF39A8
Not authorized to communications trace service tool

CPF39A9
Error occurred during communications trace function

CPF39BD
Network interface description &1 not found

CPF39B0
No communications traces exist.

CPF39B1
Trace &1 type &2 does not exist

CPF39B6
Communications trace function cannot be performed

CPF39C3
Trace &1 type &2 cannot be ended.

ENDCTLRCY (End Controller Recovery) Command Description

ENDCTLRCY Command syntax diagram

Purpose

The End Controller Recovery (ENDCTLRCY) command ends error recovery procedures for a specific controller. If any type of failure occurs after this command is run, an inquiry message is sent.

Use the Resume Controller Recovery (RSMCTLRCY) command to reestablish error recovery procedures for the controller.

Required Parameter

CTL Specifies the controller whose recovery is ended. Specify the name of the controller in the controller description.

Example for ENDCTLRCY

```
ENDCTLRCY  CTL(TROLL3)
```

This command ends error recovery procedures for the controller TROLL3.

Error messages for ENDCTLRCY

***ESCAPE Messages**

CPF2703
Controller description &1 not found.

CPF5924
Controller &1 does not allow automatic error recovery.

CPF5928
Controller &1 not varied on.

CPF5929
Controller &1 assigned to another job.

CPF5935
Error occurred during command processing.

CPF5936
Not authorized to controller &1.

ENDCPYSCN (End Copy Screen) Command Description

ENDCPYSCN Command syntax diagram

Purpose

The End Copy Screen (ENDCPYSCN) command ends the copy screen image operation. This command can be entered from any display station capable of command entry.

Note: The target display station also stops the copy screen image operation when the user presses the System Request key and types ENDCPYSCN (End Copy Screen command) on the command line. No parameters can be specified.

Optional Parameter

SRCDEV
Specifies the name of the display station that is currently having its screen images copied. If no display station name is specified, processing for the requesting display station is canceled.

***REQUESTER:** The source display station that requested the command is used.

source-device-name: Specify the display station name whose screen images are being copied.

Example for ENDCPYSCN

```
ENDCPYSCN  SRCDEV(CHARLIE)
```

The command sends a message to 'CHARLIE' (the source display station). The message indicates the copy screen image operation is about to end. The target work station display is restored to the same display image that was shown before the operation started. The sign-on display is normally shown.

Error messages for ENDCPYSCN

***ESCAPE Messages**

CPF2207
Not authorized to use object &1 in library &3 type *&2.

CPF7AF7
Device name &1 not correct.

CPF7AF8
Device name &1 not being copied.

ENDDBMON (End Database Monitor) Command Description

ENDDBMON Command syntax diagram

Purpose

The End Database Monitor (ENDDBMON) command ends the collection of database performance statistics for a specified job or all jobs on the system.

Restrictions:

1. You cannot end database monitoring for a specific job by using JOB(*ALL) on the ENDDBMON command. If JOB(*ALL) was specified on the STRDBMON command, you cannot end database monitoring for a specific job unless a STRDBMON was done on that specific job.
2. You must have job control (*JOBCTL) special authority to end a monitor for a job other than the current job.
3. For multithreaded jobs, this command is not threadsafe and may fail when the OUTFILE parameter used for the STRDBMON was a distributed file or a Distributed Data Management (DDM) file of type *SNA.

Optional Parameters

JOB Specifies the qualified name of the job and consists of as many as three elements. For example:

job-name

user-name/job-name

job-number/user-name/job-name

*N may be used in place of the user-name element to maintain position in the sequence. More information on this parameter is in Commonly used parameters.

***:** The current job's monitor is ended.

job-name: Specify the name of the job whose monitor is ended.

user-name: Specify the name of the user of the job whose monitor is ended.

job-number: Specify the number of the job whose monitor is ended.

COMMENT
Specifies the description that is associated with the database monitor record whose ID is 3018.

***BLANK:** Text is not specified.

text: Specify up to 100 characters of text.

Examples for ENDDBMON

Example 1: Ending Database Monitoring for All Jobs

```
ENDDBMON   JOB(*ALL)
```

This command ends database monitoring for all jobs on the system.

Example 2: Ending Database Monitoring for a Specific Job

```
ENDDBMON   JOB(*)
```

This command ends database monitoring for the current job.

Error messages for ENDDBMON

***ESCAPE Messages**

CPF1321
Job &1 user &2 job number &3 not found.

CPF436D
Job &1 is not being monitored.

CPF436E
Job &1 user &2 job number &3 is not active.

ENDDBG (End Debug) Command Description

ENDDBG Command syntax diagram

Purpose

The End Debug (ENDDBG) command ends debug mode for a job, removes all breakpoints and traces, clears any trace data, and removes all programs from debug mode. This command cannot be entered when one or more of the programs in the call stack are stopped at a breakpoint. All breakpoints must be canceled by Resume Breakpoint (RSMBKP) or End Request (ENDRQS) commands. After this command has been entered, all database files in production libraries can be updated normally.

If ENDDBG is not done before the job has ended, all trace data is printed.

Restriction: This command is valid only in debug mode. To start debug mode, refer to the STRDBG (Start Debug) command.

If the user is servicing another job while in debug mode, then this command must be specified before the ENDSRVJOB (End Service Job) command is allowed.

There are no parameters for this command.

Example for ENDDBG

```
ENDDBG
```

Assuming that this command is entered interactively and no program in the call stack is stopped at a breakpoint, debug mode for the job is ended.

Error messages for ENDDBG

***ESCAPE Messages**

CPF1931
Command not valid at this time.

CPF1999
Errors occurred on command.

ENDDBGSVR (End Debug Server) Command Description

ENDDBGSVR Command syntax diagram

Purpose

The End Debug Server (ENDDBGSVR) command ends the debug server router function. If there are active server jobs running when the router function is ended, the servers remain active until the connection with the client is ended. Subsequent connection requests fail until the debug server router function is started again.

There are no parameters for this command.

Example for ENDDBGSVR

```
ENDDBGSVR
```

This command ends the debug server router function.

No error messages.

ENDDEVRCY (End Device Recovery) Command Description

ENDDEVRCY Command syntax diagram

Purpose

The End Device Recovery (ENDDEVRCY) command ends error recovery procedures for a specific device. If any type of failure occurs after this command is run, an inquiry message is sent. The user must have object operational authority for the device.

Use the Resume Device Recovery (RSMDEVRCY) command to reestablish error recovery procedures for the device.

Required Parameter

DEV Specifies the device whose recovery is ended. Specify the name specified for the device in the device description.

Example for ENDDEVRCY

```
ENDDEVRCY  DEV(WSPR03)
```

This command ends error recovery procedures for the device WSPR03.

Error messages for ENDDEVRCY

***ESCAPE Messages**

CPF5923
Device &1 does not allow automatic error recovery.

CPF5925
Device &1 not varied on.

CPF5935
Error occurred during command processing.

CPF9814
Device &1 not found.

ENDDIRSHD (End Directory Shadowing) Command Description

ENDDIRSHD Command syntax diagram

Purpose

The End Directory Shadowing (ENDDIRSHD) command ends the directory shadow controlling job in the system work subsystem (QSYSWRK).

Any active collector jobs running are allowed to complete. No new collector jobs are started. Supplier jobs are prevented from starting if a collector system requests data through directory shadowing. The Start Directory Shadowing (STRDIRSHD) command can be used to re-start directory shadowing.

Restriction: You must have job control (*JOBCTL) special authority to use this command.

Optional Parameters

OPTION

Specifies whether the directory shadow controlling job is ended in a controlled manner or immediately.

***CNTRLD:** The directory shadow controlling job is ended in a controlled manner. This allows the directory shadow controlling job to perform cleanup (end-of-job processing).

***IMMED:** The directory shadow controlling job is ended immediately. The directory shadow controlling job is not allowed to perform any cleanup.

Note: Using the *IMMED option can cause unexpected results if data has been only partially updated.

DELAY

Specifies the amount of time (in seconds) allowed for the directory shadow controlling job to complete its cleanup processing during a controlled end. This parameter is not valid if OPTION(*IMMED) is specified. If the cleanup is not complete before the end of the delay time, the directory shadow controlling job is immediately ended.

30: A maximum delay time of 30 seconds is allowed for cleanup before the directory shadow controlling job is ended.

delay-time: Specify the maximum amount of delay time in seconds before the controlling job is ended. Valid values range from 1 through 999999.

Examples for ENDDIRSHD

Example 1: Ending Directory Shadowing in a Controlled Manner

```
ENDDIRSHD  OPTION(*CNTRLD)  DELAY(60)
```

The directory shadow controlling job is ended in the system work subsystem in a controlled manner and will have 60 seconds to complete its end-of-job processing.

Example 2: Ending Directory Shadowing Immediately

```
ENDDIRSHD  OPTION(*IMMED)
```

The directory shadow controlling job is ended in the system work subsystem immediately. The directory shadow controlling job does not perform end-of-job processing.

Error messages for ENDDIRSHD

***ESCAPE Messages**

CPF89A9

Unable to end job that controls directory shadowing.

ENDDSKRGZ (End Disk Reorganization) Command Description

ENDDSKRGZ Command syntax diagram

Purpose

The End Disk Reorganization (ENDDSKRGZ) command allows the user to end the disk reorganization function started using the Start Disk Reorganization (STRDSKRGZ) CL command. The user can select to end disk reorganization for all auxiliary storage pools (ASPs) or for one or more specific ASPs. A message will be sent to the system history (QHST) log when the reorganization function is ended for each ASP.

Restrictions

You must have *ALLOBJ special authority to use this command.

Optional Parameter

ASP Specifies for which auxiliary storage pool (ASP) the disk reorganization function is to be ended. A value must be specified for the ASP parameter or the ASPDEV parameter.

***ALL:** Disk reorganization will be ended for the system ASP (ASP number 1) and all basic ASPs (ASP numbers 2-32) defined to the system.

auxiliary-storage-pool-number: Specify the ASP for which disk reorganization is to be ended. Valid ASP numbers are 1 to 32. Up to 32 ASP numbers may be specified.

ASPDEV

Specifies the name of the auxiliary storage pool (ASP) device for which the disk reorganization is to be ended. A value must be specified for the ASP parameter or the ASPDEV parameter.

***ALLAVL:** Disk reorganization will be ended for all ASP devices that currently have a status of 'Available'.

auxiliary-storage-device-name: Specify the name of the independent ASP device for which disk reorganization is to be ended. Up to 32 ASP device names may be specified.

Examples for ENDDSKRGZ

Example 1: Ending Disk Reorganization for ASP 1

```
ENDDSKRGZ  ASP(1)
```

This command allows the user to end the disk reorganization function for ASP 1.

Example 2: Ending Disk Reorganization for all ASPs

```
ENDDSKRGZ  ASP(*ALL)
```

This command allows the user to end the reorganization function for each ASP that is currently being reorganized.

Example 3: Ending Disk Reorganization for all ASP Devices

```
ENDDSKRGZ  ASPDEV(*ALLAVL)
```

This command allows the user to end the reorganization function for each ASP device that is currently being reorganized.

Error messages for ENDDSKRGZ.

CPF1889

Disk reorganization not active for ASP &2.

CPF1890

*ALLOBJ authority required for requested operation.

ENDDO (End Do) Command Description

ENDDO Command syntax diagram

Purpose

The End Do (ENDDO) command is used with the DO command to identify a group of commands that are run together as a group. The ENDDO command specifies the end of the Do group that is started with an associated DO command. The ENDDO command must be specified after the last command in the Do group.

When Do groups are nested, each group must have its own ENDDO command at its end. Every ENDDO command must be associated with a DO command; if too many ENDDO commands occur in the CL program source, a message is issued and the program is not created.

Restriction: This command is valid only within a CL program.

There are no parameters for this command.

See the examples for the DO and IF commands.

Error messages for ENDDO

None

ENDGRPJOB (End Group Job) Command Description

ENDGRPJOB Command syntax diagram

Purpose

The End Group Job (ENDGRPJOB) command ends a single job in a group and resumes another job in the group. (Issue the Sign Off (SIGNOFF) command to end all jobs in a group.) The user can specify the following:

- Which job in the group is ended
- Which job in the group gains control (this is valid only when a job is ending itself)
- Whether the job ended has a job log spooled to an output queue

Note: The job that issues the ENDGRPJOB command can only end itself or some other job in the group to which this job belongs. To end a job outside the group, the END Job (ENDJOB) command must be used.

Optional Parameters

GRPJOB

Specifies the group job name of the job that is ended.

***:** This special value ends the group job that issued the ENDGRPJOB command.

group-job-name: Specify the group job name of the job that is ended.

RSMGRPJOB

Specifies the group job name of the job that is to be resumed after the active job in the group has ended. This parameter is valid only when the job that issues this command is ending itself.

*PRV: The group job most recently active is resumed.

group-job-name: Specify the group job name of the job that gains control after the active job in the group ends.

LOG Specifies whether the job log of the ending group is spooled to an output queue.

When the only job in a group is ending, one of two situations can occur:

- The ending group job is part of a secondary job pair. In this case, the other job in the secondary job pair is resumed, and control is passed back to that job.
- The ending group job is not part of a secondary job pair. In this case, the sign-on display is shown at the work station (if the work station entry for the work station device in the subsystem specifies *SIGNON).

*NOLIST: The information in the job log is deleted.

***LIST:** The information in the job log is spooled to an output queue.

Examples for ENDGRPJOB

Example 1: Ending Group Job that Issued Command

```
ENDGRPJOB  GRPJOB(*)  LOG(*PRINT)  RSMGRPJOB(GROUPJOB1)
```

This command ends the job that is currently running. Its job log is spooled to an output file for printing. When the job completes running, group job GROUPJOB1 becomes the active job in the group.

Example 2: Printing Output of Ended Job

```
ENDGRPJOB  GRPJOB(GROUPJOB2)  LOG(*PRINT)
```

Assume that the job issuing the ENDGRPJOB command is group job GROUPJOB1, which wants to end GROUPJOB2. Group job GROUPJOB2 ends. Its job log is spooled to an output file for printing.

Example 3: Ending a Job That's Part of a Secondary Job Pair

```
ENDGRPJOB  GRPJOB(*)  LOG(*NOLIST)
```

Assume that the job issuing the ENDGRPJOB command is the only job in the group and is part of a secondary job pair. The job issuing the command ends. The job's job log is not spooled to an output file. When the job ends, the other job in the secondary job pair is resumed.

Error messages for ENDGRPJOB

*ESCAPE Messages

CPF1309

Subsystem cannot complete the &1 command.

CPF1314

Value &1 for parameter &2 not allowed.

CPF1317

No response from subsystem for job &3/&2/&1.

CPF1322

The End Group Job command not allowed at this time.

CPF1323

Group job &1 not ended; parameters do not agree.

CPF1324

Group job &1 not ended; parameters do not agree.

CPF1325

Group job &1 not ended; group job &2 does not exist.

CPF1326

Group job &1 does not exist.

CPF1327

Cannot end group job &1 with ENDGRPJOB.

CPF1351

Function check occurred in subsystem for job &3/&2/&1.

ENDHOSTSVR (End Host Server) Command Description

ENDHOSTSVR Command syntax diagram

Purpose

The End Host Server (ENDHOSTSVR) command is used to end the optimized host server daemons. One or more server daemons can be ended and the server mapper daemon can be ended. » Optionally, active connections to the *DATABASE and *FILE servers can be ended with this command. «

If a server daemon is ended, and there are servers of that type that have active connections to client applications, the server jobs will remain active until communication with the client application is ended » , unless the optional ENDACTCNN parameter is specified « . Subsequent connection requests from the client application to that server daemon will fail however until the server daemon is started again.

If the server mapper daemon is ended, any existing client connections to the server jobs are unaffected. Subsequent requests from a client application to connect to the server mapper daemon (to obtain a server's port number) will fail however until the server mapper is started again.

A request to end *ALL host server daemons will end any active daemons.

» The ENDACTCNN parameter may be specified in order to end active connections to the *DATABASE and *FILE servers. This will cause the server jobs which are servicing these connections to be ended. The active connections can only be ended if the corresponding daemon job is also being ended. If the *DATABASE keyword is specified, the QZDASOINIT and QZDASSINIT jobs which have active connections will be ended. If the *FILE keyword is specified, the QPWFSERVSO and QPWFSERVSS jobs which have active connections will be ended. «

Required Parameter

SERVER

Specifies the server daemons to be ended.

***ALL:** All of the server daemons and the server mapper daemon are ended.

***CENTRAL:** The central server daemon in the QSYSWRK subsystem, if active, is ended.

***DATABASE:** The database server daemon in the QSERVER subsystem, if active, is ended.

***DTAQ:** The data queue server daemon in the QSYSWRK subsystem, if active, is ended.

***FILE:** The file server daemon in the QSERVER subsystem, if active, is ended.

***NETPRT:** The network print server daemon in the QSYSWRK subsystem, if active, is ended.

***RMTCMD:** The remote command and distributed program call server daemon in the QSYSWRK subsystem, if active, is ended.

***SIGNON:** The signon server daemon in the QSYSWRK subsystem, if active, is ended.

***SVRMAP:** The server mapper daemon in the QSYSWRK subsystem, if active, is ended.

» Optional Parameter

ENDACTCNN

Specifies whether or not the active connections for the specified servers will be ended.

Single Value

***NONE:** No active connections will be ended.

Specific Server Values

***DATABASE:** The active connections being serviced by the QZDASOINIT and QZDASSINIT server jobs will be ended. The server jobs servicing these connections will be ended.

***FILE:** The active connections being serviced by the QPWFSERVSO and QPWFSERVSS server jobs will be ended. The server jobs servicing these connections will be ended. «

Examples for ENDHOSTSVR

Example 1: Ending All Host Server Daemons

```
ENDHOSTSVR   SERVER(*ALL)
```

This command ends all active server daemons and the server mapper daemon. Any active connections to client applications with the server jobs are unaffected. Ending all host server daemons prevents any subsequent client connection requests from succeeding.

Example 2: Ending Specific Server Daemons

```
ENDHOSTSVR   SERVER(*CENTRAL *SVRMAP)
```

This command ends the central server daemon and the server mapper daemon. Both daemon jobs run in the QSYSWRK subsystem.

» Example 3: Ending Specific Server Daemons and Active Connections

```
ENDHOSTSVR   SERVER(*CENTRAL *DATABASE) ENDACTCNN(*DATABASE)
```

This command ends the central server daemon in the QSYSWRK subsystem and the database server daemon in the QSERVER subsystem. Additionally, the active connections to the *DATABASE server, and the QZDASOINIT and QZDASSINIT server jobs which are servicing these connections, will be ended. «

ENDHTTPCRL (End HTTP Crawl) Command Description

Note: To use this command, you must have the 5722-DG1 (HTTP Server) product installed.

Purpose

The End HTTP (ENDHTTPCRL) command stops or pauses an active crawling session. Only a document list that is currently being updated by a crawling action is allowed to end.

Restriction: You must have *IOSYSCFG special authority to use this command.

Parameters

OPTION

Specifies the action to perform.

The possible values are:

***STOP**

Stop an active crawling session.

***PAUSE**

Pause an active crawling session.

DOCLIST

Specifies the document list file where the list of path names are stored.

Error messages for ENDHTTPCRL

***ESCAPE messages**

HTP165E

Request to pause or end a crawling session failed. Reason &1.

ENDINP (End Input) Command Description

ENDINP Command syntax diagram

Purpose

The End Input (//ENDINP) command is a delimiter that indicates the end of the input in a batch input stream. The //ENDINP command also can indicate the end of an inline data file if the command is detected while the inline file is being processed. If the inline file is using ending characters which are not defaults, the //ENDINP command is embedded without being recognized.

Restriction: The ENDINP command cannot be used from a work station. Two slashes must precede this command name when entering it in the data record, that is, //ENDINP. The user can separate the slashes from this command name with blank spaces, for example, //   ENDINP.

There are no parameters for this command.

Example for ENDINP

```
//BCHJOB
*
*
*
//DATA
*
*
*
//ENDINP
```

The ENDINP command indicates the end of a input stream that began with the Batch Job (BCHJOB) command.

Error messages for ENDINP

***ESCAPE Messages**

CPF1753
Command cannot be run.

ENDIPSIFC (End IP over SNA Interface) Command Description

ENDIPSIFC Command syntax diagram

Purpose

The End IP over SNA Interface (ENDIPSIFC) command is used to end an AF_INET sockets over SNA interface (an IP address by which this local host is known on the SNA transport).

Note: Ending an interface causes all routes associated with this interface to be deactivated immediately unless there are other active interfaces that the routes can switch to.

Required Parameter

INTNETADR
Specifies the internet address of an active (started) interface that had previously been added to the IP SNA configuration with the Add IP over SNA Interface (ADDIPSIFC) CL command. The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. If the internet address is entered from a command line, the address must be enclosed in apostrophes.

Example for ENDIPSIFC

```
ENDIPSIFC   INTNETADR('9.5.1.248')
```

This command deactivates (ends) the interface with IP address 9.5.1.248 and

Error messages for ENDIPSIFC

***ESCAPE Messages**

CPFA114
IP over SNA interface &1 not ended.

ENDJOB (End Job) Command Description

ENDJOB Command syntax diagram

Purpose

The End Job (ENDJOB) command ends the specified job and any associated inline data files. The job may be on a job queue, it may be active within a system, or it may have already completed running.

You can specify that the application program is given time to control end-of-job processing. If no time is given or if cleanup cannot be performed within the given time, the system performs minimal end-of-job processing, which can include:

- Closing the database files.
- Spooling the job log to an output queue.
- Cleaning up internal objects in the operating system.
- Showing the end-of-job display (for interactive jobs).
- Completing commitment control processing

Before ending the job, you should verify that no logical unit of work is in an in doubt state due to a two-phase commit operation that is in progress. If it is, then the value of the Action if ENDJOB commitment option can greatly impact the ENDJOB processing. For example, if the Action if ENDJOB commitment option is the default value of WAIT, this job will be held up and will not complete its end of job processing until the commitment control operation is completed. This ensures database integrity on all related systems. For specific instructions on how to determine these conditions, and for a description of all the impacts of ending this job under these conditions, see the Backup, Recovery, and Availability topic in the Information Center.

All spooled files associated with the job being ended can also be deleted or allowed to remain on the output queue.

Restrictions:

1. The issuer of the command must be running under a user profile which is the same as the job user identity of the job being ended, or the issuer of the command must be running under a user profile which has job control (*JOBCTL) special authority.

   The job user identity is the name of the user profile by which a job is known to other jobs. It is described in more detail in the Work Management book.

Required Parameter

JOB Specifies the name of the job that is ended. If no job qualifier is given, all of the jobs currently in the system are searched for the simple job name. If more than one of the specified name is found, a qualified job name must be specified.

A job identifier is a qualified name with up to three elements. For example:

```
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Commonly used parameters.

job-name: Specify the name of the job that is ended.

user-name: Specify the name of the user of the job that is ended.

job-number: Specify the number of the job that is ended.

Optional Parameters

OPTION

Specifies whether the job is ended in a controlled manner, which lets the application program perform end-of-job processing, or is ended immediately. In either case, the system performs certain job cleanup processing.

***CNTRLD:** The job is ended in a controlled manner. This allows the program to perform cleanup (end-of-job processing). The application has the amount of time specified on the DELAY parameter to complete cleanup before the job is ended.

***IMMED:** The job ends immediately and the system performs end-of-job cleanup. System cleanup can take from a brief amount of time to several minutes.

Note: This value is recommended only if specifying the *CNTRLD value has been unsuccessful. When you specify the *IMMED value, you can get undesirable results, for example, from data that has been partially updated.

DELAY

Specifies the amount of time (in seconds) that the job has to complete its end-of-job processing during a controlled end. If the end-of-job processing is not completed before the end of the delay time, the job is ended immediately, and only system cleanup is performed.

The delay time does not start until the job becomes active if the job is suspended because of one of the following conditions:

- The system request option 1 is selected.
- The job is held by the Hold Job (HLDJOB) command.
- The job is transferred by the Transfer Secondary Job (TFRSECJOB) command.
- The job is transferred by the Transfer Group Job (TFRGRPJOB) command.

Note: This parameter is valid only when OPTION(*CNTRLD) is specified.

30: A maximum delay time of 30 seconds is allowed for cleanup before the job is ended.

delay-time: Specify the maximum amount of delay time (in seconds) before the job is ended. Valid values range from 1 through 999999 seconds. For additional information on the use of the DELAY parameter for ending an end-of-file delay job, refer to the EOFDLY (End of File Delay) parameter of the Override Database File (OVRDBF) command.

SPLFILE

Specifies whether to delete the spooled files created by this job. Regardless of whether the spooled files are deleted, the job logs related to the spooled files are kept.

***NO:** The spooled files are not deleted. They are kept for normal processing by a writer. » When the job ends, the spooled file action (SPLFACN) job attribute determines whether spooled files are detached from the job or kept with the job. «

***YES:** The spooled files are deleted. » If the job has already ended and the spooled file action for the job is to detach the spooled files, the End Job (ENDJOB) command will not find the job and the spooled files will not be deleted. «

LOGLMT

Specifies the logging limit, which is the maximum number of message entries that are written to the job log printer file (QPJOBLOG) from the message queue of an ending job.

If a job to be ended with this command is already ending, the value specified on this parameter can change the logging limit of the job that is ending. The following are examples of how the logging limit can be changed:

1. If the value specified is greater than the number of messages written at the time the command is issued, messages continue to be written until the new limit is reached.
2. If the value specified is less than the number of messages already written to the spooled file, a message indicating that the limit has been reached is immediately put in the spooled file as the last entry. The remaining messages on the queue are ignored.
3. If 0 (zero) is specified before any messages are written to the spooled file, no job log is produced for the job that is ending.

***SAME:** The value does not change. If no logging limit has been established previously for the job, the system uses the *NOMAX value.

***NOMAX:** There is no maximum number of message entries logged. All messages on the job message queue are written to the job log.

maximum-logged-entries: Specify the maximum number of messages written to the job log.

ADLINTJOBS

Specifies whether additional interactive jobs associated with the job specified in the JOB parameter are being ended. The additional interactive jobs can be group jobs or all jobs associated with the work station (group and secondary jobs) where the job specified in the JOB parameter is running. The job specified in the JOB parameter does not have to be the active job. A job name entered on the JOB parameter must resolve to a single interactive job for this parameter to be used. If the job is not an interactive job, an error message is sent.

***NONE:** Only the job specified in the JOB parameter is ended.

***GRPJOB:** If the job specified in the JOB parameter is a group job, all jobs associated with the group are ended. If the job is not a group job, the job specified in the JOB parameter is ended.

***ALL:** All interactive jobs running on the work station associated with the job specified in the JOB parameter are ended, including group jobs and secondary jobs.

DUPJOBOPT

Specifies the action taken when duplicate jobs are found by this command.

***SELECT:** The selection display is shown when duplicate jobs are found during an interactive session. Otherwise, a message is issued.

***MSG:** A message is issued when duplicate jobs are found.

Examples for ENDJOB

Example 1: Ending a Job Immediately

```
ENDJOB   JOB(JOB1)  OPTION(*IMMED)  SPLFILE(*YES)
```

This command ends a job named JOB1 immediately. Spooled output produced by the job is deleted; the job log is saved.

Example 2: Saving Spooled Output

```
ENDJOB   JOB(001234/XYZ/JOB2)  OPTION(*CNTRLD)
  DELAY(50)  SPLFILE(*NO)
```

This command ends a job named 001234/XYZ/JOB2. Spooled output is saved for normal processing by the spooling writer. The job has 50 seconds to perform any cleanup routines, after which it is ended immediately.

Error messages for ENDJOB

*ESCAPE Messages

CPF1317

No response from subsystem for job &3/&2/&1.

CPF1321

Job &1 user &2 job number &3 not found.

CPF1332

End of duplicate job names.

CPF1340

Job control function not performed.

CPF1341

Reader or writer &3/&2/&1 not allowed as job name.

CPF1342

Current job not allowed as job name on this command.

CPF1343
Job &3/&2/&1 not valid job type for function.

CPF1344
Not authorized to control job &3/&2/&1.

CPF1351
Function check occurred in subsystem for job &3/&2/&1.

CPF1352
Function not done. &3/&2/&1 in transition condition.

CPF1359
ENDJOBABN not allowed at this time for job &3/&2/&1.

CPF1360
&3/&2/&1 already ending because of ENDJOBABN.

CPF1361
Job &3/&2/&1 already ending with *IMMED option.

CPF1362
Job &3/&2/&1 has completed.

CPF1363
Job &3/&2/&1 is already ending *CNTRLD.

CPF8172
Spool control block for job &10/&9/&8 damaged.

ENDJOBABN (End Job Abnormal) Command Description

ENDJOBABN Command syntax diagram

Purpose

The End Job Abnormal (ENDJOBABN) command ends a job that cannot be ended successfully by running the End Job (ENDJOB) command with OPTION(*IMMED) specified. The ENDJOBABN command cannot be issued against a job until 10 minutes have passed following the request for immediate ending. This allows sufficient time for normal job ending functions to occur.

When the ENDJOBABN command is issued, most of the end-of-job processing is bypassed (including spooling of the job log, the end of job display for interactive jobs, and the end-of-job processing for the specific functions that are being performed). The part of the end-of-job processing that is attempted is allowed only five minutes to complete. If it does not do so in five minutes, the job is forced to end at that point. Because some of the job cleanup is not performed, the ENDJOBABN command should only be used when a job that is in the process of immediate ending does not finish ending and resources in use by the job are needed by another job or by the system. When the ENDJOBABN command is used, some resources in use by the ended job may be left unavailable until the next IPL.

Use of the ENDJOBABN command causes the next system end to be marked as ABNORMAL. Certain system functions are then called during the subsequent IPL to clear up conditions that may have resulted from running the ENDJOBABN command. This does not, however, cause any machine recovery functions to be called, nor do any access paths need to be rebuilt. Some storage in use by the job may become unavailable after the ENDJOBABN command is run and that available storage can be reclaimed by using the Reclaim Storage (RCLSTG) command.

Bypassing the job log writing process causes the job to have the status of JOBLOG PENDING (as shown on the DSPJOB status attributes display) after it has been ended with the ENDJOBABN command. The job log writing is not performed until the next IPL. However, the contents of the job log can be printed or shown by using the Display Job Log (DSPJOBLOG) command.

When the ENDJOBABN command is run, the following functions are performed successfully:

- Journaling entries
- Commitment control

  Before ending the job abnormally, you should verify that no logical unit of work is in an in doubt state due to a two-phase commit operation that is in progress. If it is, then pending committable changes at this system will **not** be committed or rolled back. Therefore, database integrity may not be maintained on all related systems. For specific instructions on how to determine these conditions, and for a description of all the impacts of ending this job abnormally under these conditions, see the Commitment control article in the Information Center.
- Making database files available for use by others
- Releasing file locks

This command fails to end a job or takes more than five minutes to do so in the following situations:

- When the job runs under a subsystem monitor that is hung, is abnormally slow, or has ended abnormally (the subsystem monitor performs part of the ending function).
- When the machine interface (MI) instruction running in the job is hung or is abnormally slow. The job cannot end until the MI instruction that is currently running completes or reaches a point of interruption.

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, and QSRV user profiles have private authorities to use the command.
2. The issuer of the command must be running under a user profile which is the same as the job user identity of the job being ended, or the issuer of the command must be running under a user profile which has job control (*JOBCTL) special authority.

   The job user identity is the name of the user profile by which a job is known to other jobs. It is described in more detail in the Work Management book.
3. After the ENDJOBABN command is run, subsequent ENDJOBABN commands cannot be issued against the job.
4. Users cannot end a reader, writer, subsystem monitor, or system job.
5. Users cannot run the ENDJOBABN command until ten minutes *after* immediate ending of the job is started. Immediate ending of the job is started in the following ways:
   - When the End Job (ENDJOB) command with OPTION(*CNTRLD) is specified and the delay time ends
   - When the End Job (ENDJOB) command with OPTION(*IMMED) is issued
   - When the End Subsystem (ENDSBS) command with OPTION(*CNTRLD) is issued against the subsystem in which the job is running and the delay time ends
   - When the End Subsystem (ENDSBS) command with OPTION(*IMMED) is issued against the subsystem in which the job is running
   - When the End System (ENDSYS) command with OPTION(*IMMED) is issued, or OPTION(*CNTRLD) is issued and the delay time ends

Required Parameter

JOB Specifies the name of the job to be ended. If no job qualifier is given, all of the jobs currently in the system are searched for the simple job name. If more than one of the specified names are found, a qualified job name must be specified to distinguish them.

A job identifier is a qualified name with up to three elements. For example:

```
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Commonly used parameters.

job-name: Specify the name of the job to be ended.

user-name: Specify the name of the user of the job to be ended.

job-number: Specify the number of the job to be ended.

Optional Parameter

DUPJOBOPT
Specifies the action taken when duplicate jobs are found by this command.

***SELECT:** The selection display is shown when duplicate jobs are found during an interactive session. Otherwise, a message is issued.

***MSG:** A message is issued when duplicate jobs are found.

Example for ENDJOBABN

```
ENDJOBABN  JOB(000310/SMITH/PAYROLL)
```

This command ends the batch job 000310/SMITH/PAYROLL after the failure of an earlier attempt to end it with the ENDJOB command. The ENDJOBABN command can be issued only after waiting at least ten minutes for the job to end after issuing the ENDJOB command.

Error messages for ENDJOBABN

***ESCAPE Messages**

CPF1317
No response from subsystem for job &3/&2/&1.

CPF1321
Job &1 user &2 job number &3 not found.

CPF1332
End of duplicate job names.

CPF1340
Job control function not performed.

CPF1341
Reader or writer &3/&2/&1 not allowed as job name.

CPF1342
Current job not allowed as job name on this command.

CPF1343
Job &3/&2/&1 not valid job type for function.

CPF1351
Function check occurred in subsystem for job &3/&2/&1.

CPF1359
ENDJOBABN not allowed at this time for job &3/&2/&1.

CPF1360
&3/&2/&1 already ending because of ENDJOBABN.

CPF1362
Job &3/&2/&1 has completed.

ENDJS (End Job Scheduler) Command Description

Note: To use this command, you must have the 5722-JS1 (Job Scheduler for iSeries) licensed program installed.

ENDJS Command syntax diagram

Purpose

The End Job Scheduler (ENDJS) command allows you to end the Job Scheduler job monitor or to end the capture of job information for application software that you started using the STRJS command.

Optional Parameter

OPTION
Specifies whether you want to end the Job Scheduler monitor or you want to stop capturing job information.

***MONITOR:** End the Job Scheduler monitor.

***CAPTURE:** Stop capturing job information from application software.

Example for ENDJS

Example 1: Ending the Job Scheduler Monitor

```
ENDJS OPTION(*MONITOR)
```

In this example the Job Scheduler monitor is ended. The monitor stays inactive until you process the Start Job Scheduler (STRJS) command.

Error messages for ENDJS

None

ENDJOBTRC (End Job Trace) Command Description

Note: To use this command, you must have the 5722-PT1 (Performance Tools for iSeries) licensed program installed.

ENDJOBTRC Command syntax diagram

Purpose

The End Job Trace (ENDJOBTRC) command turns off the OS/400 system job tracing function that was started by the Start Job Trace (STRJOBTRC) command, saves all collected trace records in a database file, and optionally produces reports used to analyze the data from a performance viewpoint. The Print Job Trace (PRTJOBTRC) command may also be used to produce reports from the same source data.

Optional Parameters

MBR Specifies the member in file QAPTTRCJ in which to save the trace data. If the member does not exist, it is added to the file. If the member exists, the new data replaces the old contents.

QAJOBTRC: The standard member name, QAJOBTRC, is used.

member-name: Specify the name of an alternative member in which to save the data. This allows several sets of job trace data to be kept.

LIB Specifies the library where the trace data is saved. If the library does not contain a file named QAPTTRCJ, one is created.

QPFRDATA: The trace data is stored in the IBM-supplied performance data library, QPFRDATA.

library-name: Specify the name of an alternative library where the trace data is saved. This is necessary if access to the default library is not allowed.

RPTTYPE
Specifies the type of reports to produce.

***NONE:** No reports are produced; however, the Print Job Trace (PRTJOBTRC) command may be used later to report on the saved data. If this value is selected, the remaining parameters are not applicable.

***DETAIL:** A report is produced that shows the individual job trace records in detail. The output is directed to the printer file QPPTTRCD. Each page heading includes the text 'Job Trace Information'.

***SUMMARY:** Two reports are produced that summarize the job trace data by work station transaction. One report shows primarily physical disk activity; its printer file is QPPTTRC1, and its page heading includes the text 'Trace Analysis Summary'. The other report concentrates on higher level activities such as database I/O and inter-program transfers of control; its printer file is QPPTTRC2, and its page heading includes the text 'Trace Analysis I/O Summary'.

***BOTH:** Both the detail and summary reports are produced (three reports total).

TITLE Specifies a title that is printed on the page heading of each report.

***BLANK:** A blank title is used.

'report-title': Specify any title of up to 50 characters, enclosed in apostrophes. This may be used, for example, to distinguish between reports on different sets of trace data or different sections of the same data.

STRSEQ
Specifies the sequence number of the first job trace record to include in any reports. No records before this one are listed in the detail report or counted in either summary report.

***FIRST:** Trace records starting from the first one (sequence number 1) are included.

sequence-number: Specify the sequence number of the first trace record to include. A value is determined by previewing reports produced from all the job trace data. This can be used to bracket a particular set of transactions on which to report.

ENDSEQ
Specifies the sequence number of the last job trace record to include in any reports. No records following this one are listed in the detail report or counted in either summary report.

***LAST:** Trace records through the last one are included.

sequence-number: Specify the sequence number of the last trace record to include. As with STRSEQ, a value can be chosen through a preview process so as to bracket a particular set of transactions.

ENDTNS
Specifies the program that signifies the end of a transaction (when followed by the STRTNS parameter).

QT3REQIO: The low-level OS/400 system work station I/O program, QT3REQIO, is used. This value is used to break the trace data into work station transactions.

program-name: Specify the name of the program that ends a transaction. This allows reporting on transactions that do not involve work stations, such as communications lines.

STRTNS

Specifies the program that signifies the start of a transaction (when preceded by the program specified in ENDTNS parameter).

QWSGET: The OS/400 system work station input program, QWSGET, is used. This value is used to break the trace data into work station transactions.

program-name: Specify the name of the program that starts a transaction.

JOB Specifies the job name used if the job is submitted for batch processing.

Note: If *NONE is specified on the JOBD parameter, this parameter is ignored; job processing is performed interactively.

ENDJOBTRC: The command name is used for the job name.

***MBR:** The name specified for the performance data member on the MBR parameter is used.

job-name: Specify the name used for batch jobs.

JOBD Specifies the job description used to submit jobs for batch processing.

The name of the job description can be qualified by one of the following library values:

- ***LIBL:** All libraries in the job's library list are searched until the first match is found.
- ***CURLIB:**The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.
- *library-name:*Specify the name of the library to be searched.

QPFRJOBD: The IBM-supplied Performance Tools job description, QPFRJOBD, is used.

job-description-name: Specify the name of an alternative job description.

Other Single Values

***NONE:** A batch job is not submitted; instead, processing continues interactively while the user waits. The user's work station is not available for other use during this time, which could be significant for long jobs.

Examples for ENDJOBTRC

Example 1: Stopping Job Tracing

```
ENDJOBTRC
```

This command stops tracing and saves the created trace records in QPFRDATA/QAPTTRCJ, member QAJOBTRC. No reports are produced.

Example 2: Producing a Report of Job Records

```
ENDJOBTRC  LIB(MYLIB)  RPTTYPE(*DETAIL)
```

This command stops job tracing, saves the created trace records in member QAJOBTRC of MYLIB/QAPTTRCJ, and produces a detail report. The use of a private library allows several users to trace jobs at the same time.

Error messages for ENDJOBTRC

None.

ENDJRN (End Journal) Command Description

ENDJRN Command syntax diagram

Purpose

The End Journal (ENDJRN) command is used to end the journaling of changes for an object or list of objects. The object types which are supported through this interface are Data Areas (*DTAARA), Data Queues (*DTAQ), Byte Stream Files (*STMF), Directories (*DIR), and Symbolic Links (*SYMLNK). Only objects of type *STMF, *DIR or *SYMLNK that are in the Root ('/'), QOpensys, and User-defined file systems are supported.

All objects of the supported types that are currently being journaled to a specific journal may also have journaling stopped.

Note: For other ways to end journaling see the following commands:
- Access Paths - End Journal Access Path (ENDJRNAP)
- Physical Files - End Journal Physical File (ENDJRNPF)
- Other Objects - End Journal Object (ENDJRNOBJ)

Restrictions:
1. Objects specified on the command cannot be in use for any reason at the time the command is running.
2. If OBJ(*ALL) is specified, a journal name must be specified (JRN parameter).
3. If a journal name and a list of object names are specified, all objects must be currently journaled to the indicated journal.
4. The specified journal cannot be a remote journal.
5. At least one of parameter OBJ or OBJFID must be specified.

Required Parameters

OBJ Specifies a maximum of 300 object path names for which changes will no longer be journaled. Only objects whose path name identifies an object of type *STMF, *DIR, *SYMLNK, *DTAARA or *DTAQ are supported.

Single value

***ALL:** All objects of the supported type that are currently being journaled to the indicated journal are to stop having changes journaled. If *ALL is specified, parameter OBJFID must not be specified.

Element 1: Object Name

'object-path-name': Specify the path name of the object for which changes are no longer journaled.

A pattern can be specified in the last part of the path name. An asterisk (*) matches any number of characters and a question mark (?) matches a single character. If the path name is qualified or contains a pattern it must be enclosed in apostrophies. Symbolic links within the path name will not be followed. If the path name begins with the tilde (~) character, then the path is assumed to be relative to the appropriate home directory.

Additional information about path name patterns is in the Integrated File System Introduction topic in the Information Center.

Element 2: Include or Omit

The second element specifies whether names that match the pattern should be included or omitted from the operation. Note that in determining whether a name matches a pattern, relative name patterns are always treated as relative to the current working directory.

***INCLUDE:** The objects that match the object name pattern are to stop having their changes journaled unless overridden by an *OMIT specification.

***OMIT:** The objects that match the object name pattern are not to be included with the objects that are to stop having their changes journaled. This overrides an *INCLUDE specification and is intended to be used to omit a subset of a previously selected path.

OBJFID

Specifies a maximum of 300 file identifiers (FID) for which changes are no longer to be journaled. FIDs are a unique identifier associated with integrated file system related objects. This field is input in hexadecimal format. Only objects whose FID identifies an object of type *STMF, *DIR, *SYMLNK, *DTAARA or *DTAQ are supported.

file-identifier: Objects identified with the FID are no longer journaled.

Optional Parameters

JRN Specifies the path name of the journal to which changes are currently being journaled.

***OBJ:** The path name of the journal is determined by the system from the specified object path name or object file identifier.

journal-path-name: Specify the path name of the journal to which changes are currently being journaled.

SUBTREE

Specifies whether the objects in directory subtrees are to stop having their changes journaled.

Note: This parameter is ignored if the OBJFID parameter is specified.

***NONE:** Only the objects that match the selection criteria are processed. The objects within selected directories are not implicitly processed.

***ALL:** All objects that meet the selection criteria are processed in addition to the entire subtree of each directory that matches the selection criteria. The subtree includes all subdirectories and the objects within those subdirectories.

PATTERN

Specifies a maximum of 20 patterns to be used to include or omit objects for the end journal operation.

Note: This parameter is ignored if the OBJFID parameter is specified.

Element 1: Name Pattern

'*': All objects that match the input OBJ parameter are to be included into the end journal operation or omitted from the end journal operation.

name-pattern: Specify the pattern to either include or omit objects for the end journal operation. Only the last part of the path name will be considered for the name pattern match. Path name delimiters are not allowed in the name pattern.

If the Name Pattern parameter is not specified the default will be to match all patterns.

A pattern can be specified in the last part of the path name. An asterisk (*) matches any number of characters and a question mark (?) matches a single character. If the path name is qualified or contains a pattern it must be enclosed in apostrophies. Symbolic links within the path name will not be followed. If the path name begins with the tilde (~) character, then the path is assumed to be relative to the appropriate home directory.

Additional information about path name patterns is in the Integrated File System Introduction topic in the Information Center.

Element 2: Include or Omit

The second element specifies whether names that match the pattern should be included or omitted from the operation. Note that in determining whether a name matches a pattern, relative name patterns are always treated as relative to the current working directory.

Note: The SUBTREE parameter specifies whether directory subtrees are included or omitted.

***INCLUDE:** The objects that match the object name pattern are to stop having their changes journaled unless overridden by an *OMIT specification.

***OMIT:** The objects that match the object name pattern are not to be included with the objects that are to stop having their changes journaled. This overrides an *INCLUDE specification and is intended to be used to omit a subset of a previously selected pattern.

Examples for ENDJRN

Example 1: End All Non-Database Journaling

```
ENDJRN  OBJ(*ALL) JRN('/qsys.lib/mylib.lib/myjrn.jrn')
```

This command stops the journaling of all changes to all objects of type *DIR, *STMF, *SYMLNK, *DTAARA and *DTAQ to journal /qsys.lib/mylib.lib/myjrn.jrn.

Example 2: End Journaling with Omit of Directory

```
ENDJRN  OBJ(('/mypath' *INCLUDE) ('/mypath/myobject' *OMIT))
```

This command stops the journaling of all changes to all first-level objects in directory /mypath except object /mypath/myobject. Object /mypath/myobject will continue to be journaled.

Example 3: End Journaling with Pattern Selection

```
ENDJRN  OBJ(('/mypath' *INCLUDE) ('/mypath/mysubdir' *OMIT))
        SUBTREE(*ALL) PATTERN(('*.txt' *INCLUDE))
```

This command stops the journaling of all changes to all objects in directory /mypath of type *DIR, *STMF, and *SYMLNK that match pattern '*.txt'. Any objects within directory /mypath/mysubdir will continue to be journaled.

Example 4: End Journaling using File Identifiers

```
ENDJRN  OBJFID(00000000000000007E09BDB000000009
               00000000000000009E09BDB00000000A)
```

This command stops the journaling of all changes to the objects of type *DIR, *STMF, *SYMLNK, *DTAARA or *DTAQ represented by the specified file identifiers.

Example 5: End Journaling on a set of Data Areas

```
ENDJRN OBJ(('/qsys.lib/mylib.lib/mydata*.dtaara'))
       JRN('/qsys.lib/mylib.lib/myjrn.jrn')
```

This command stops the journaling of all changes to the objects of type *DTAARA in library MYLIB that begin with the characters 'MYDATA'.

Error messages for ENDJRN

***ESCAPE Messages**

CPFA0D4
File system error occurred.

CPF700B
&1 of &2 objects have ended journaling.

CPF705A
Operation failed due to remote journal.

CPF9801
Object &2 in library &3 not found.

CPF9802
Not authorized to object &2 in &3.

CPF9803
Cannot allocate object &2 in library &3.

CPF9810
Library &1 not found.

CPF9820
Not authorized to use library &1.

» **CPF9825**
Not authorized to device &1.«

CPF9830
Cannot assign library &1.

» **CPF9873**
ASP status is preventing access to object.

CPF9875
Resources exceeded on ASP &1.«

ENDJRNAP (End Journal Access Path) Command Description

ENDJRNAP Command syntax diagram

Purpose

The End Journal Access Path (ENDJRNAP) command is used to end journaling of the access paths of a file.

All access paths currently being journaled to a specific journal may also have journaling stopped.

Note: For other ways to end journaling see the following commands:

- Integrated File System objects - End Journal (ENDJRN)
- Physical Files - End Journal Physical File (ENDJRNPF)
- Other objects - End Journal Object (ENDJRNOBJ)

Restrictions:

1. The access paths for the files specified on the command cannot be in use for any reason at the time the command is running.
2. Overrides are not applied to the files listed in the FILE parameter.

3. If FILE(*ALL) is specified, a journal name must be specified.
4. If a journal name and a list of file names are specified, then all the access paths for the listed files must be currently journaled to the indicated journal.
5. Journaling entries for any physical file does not end by the running of this command.
6. This command cannot be used on or with a remote journal.

Required Parameter

FILE Specifies a maximum of 50 qualified names of the database files for which journaling for their access paths is ended.

***ALL:** Journaling ends for all access paths that are currently being journaled to the specified journal.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specifies the name of the database file that has journaling ended for its access paths.

Optional Parameter

JRN Specifies the qualified name of the journal to which the access paths for the indicated files are currently being journaled.

***FILE:** The journal is determined by the system from the specified file names.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

journal-name: Specifies the name of the journal to which the access paths for the indicated files are currently being journaled.

Example for ENDJRNAP

```
ENDJRNAP   FILE(MYLIB/MYFILE)
```

This command ends the journaling for all access paths of the file MYFILE in the library MYLIB.

Error messages for ENDJRNAP

*ESCAPE Messages

CPF6972
Cannot allocate access path for file &1 in &2.

CPF7008
Cannot start or end access path journaling for file &1.

CPF703C
DDL transaction prevents journaling operation.

CPF703D
DDL transaction prevents journaling operation.

CPF703E
DDL transaction prevents journaling operation.

CPF7032
ENDJRNPF or ENDJRNAP command failed.

CPF7033
Start or end journaling failed for member &3.

CPF7034
Logical damage of file &1 in &2.

CPF705A
Operation failed due to remote journal.

CPF708D
Journal receiver found logically damaged.

CPF9801
Object &2 in library &3 not found.

CPF9802
Not authorized to object &2 in &3.

CPF9803
Cannot allocate object &2 in library &3.

CPF9812
File &1 in library &2 not found.

CPF9820
Not authorized to use library &1.

CPF9822
Not authorized to file &1 in library &2.

» **CPF9825**
Not authorized to device &1.«

CPF9830
Cannot assign library &1.

» **CPF9873**
ASP status is preventing access to object.

CPF9875
Resources exceeded on ASP &1.«

ENDJRNOBJ (End Journal Object) Command Description

ENDJRNOBJ Command syntax diagram

Purpose

The End Journal Object (ENDJRNOBJ) command is used to end journaling of changes for an object or list of objects.

All objects, of object types *DTAARA and *DTAQ, that are currently being journaled to a specific journal may also have journaling stopped.

Note: To end journaling for other object types, see the following commands:

- Access Paths - End Journal Access Path (ENDJRNAP)
- Integrated File System objects - End Journal (ENDJRN)
- Physical Files - End Journal Physical File (ENDJRNPF)

Restrictions:

1. Objects specified on the command cannot be in use for any reason at the time the command is running.
2. If OBJ(*ALL) or OBJTYPE(*ALL) is specified, a journal name must be specified (JRN parameter).
3. If a journal name and a list of object names are specified, all objects must be currently journaled to the indicated journal.
4. The specified journal cannot be a remote journal.

Optional Parameters

OBJ Specifies a maximum of 300 qualified object names for which changes are no longer to be journaled.

Single Value

***ALL:** All objects of the specified object types that are currently being journaled to the indicated journal are to stop having their changes journaled.

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the name of the object for which journaling is to be ended.

OBJTYPE

Specifies the object type for which changes are to be ended.

***ALL:** All objects of the object types that are supported on this command are to have their journaling ended.

***DTAARA:** Data area objects are to have their journaling ended.

***DTAQ:** Data queue objects are to have their journaling ended.

JRN Specifies the qualified name of the journal to which changes in the objects are currently being journaled.

***OBJ:** The journal is determined by the system from the specified object name and object type.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

journal-name: Specify the name of the journal to which the indicated objects are currently being journaled.

Examples for ENDJRNOBJ

Example 1: End Journaling All Data Areas and Data Queues

```
ENDJRNOBJ  OBJ(*ALL) OBJTYPE(*ALL) JRN(MYLIB/MYJRN)
```

This command stops journaling all changes to all objects of type *DTAARA and *DTAQ to journal MYJRN in library MYLIB.

Example 2: End Journaling for Specific Data Area

```
ENDJRNOBJ  OBJ(DTALIB/MYDTAARA) OBJTYPE(*DTAARA)
```

This command stops the journaling of all changes to data area MYDTAARA in library DTALIB.

Error messages for ENDJRNOBJ

***ESCAPE Messages**

CPF700B
&1 of &2 objects have ended journaling.

CPF705A
Operation failed due to remote journal.

CPF9801
Object &2 in library &3 not found.

CPF9802
Not authorized to object &2 in &3.

CPF9803
Cannot allocate object &2 in library &3.

CPF9810
Library &1 not found.

CPF9820
Not authorized to use library &1.

» **CPF9825**
Not authorized to device &1.«

CPF9830
Cannot assign library &1.

» **CPF9873**
ASP status is preventing access to object.

CPF9875
Resources exceeded on ASP &1.«

ENDJRNPF (End Journal Physical File) Command Description

ENDJRNPF Command syntax diagram

Purpose

The End Journal Physical File (ENDJRNPF) command is used to end journaling of changes for a specific physical file and all of its members.

All physical files currently being journaled to a specific journal may also have journaling stopped.

When the file for which journaling is ended is a distributed file, an attempt is made to distribute the ENDJRNPF command if journaling was successfully ended locally. Even if the distribution request fails, the local file is not journaled. In addition, if a journal and file name are specified, and the file is distributed, an attempt to distribute the ENDJRNPF request is made even if the file is not journaled locally.

Note: For other ways to end journaling see the following commands:

- Access Paths - End Journal Access Path (ENDJRNAP)
- Integrated File System objects - End Journal (ENDJRN)

Restrictions:

1. Members in the files specified on the command cannot be in use for any reason at the time the command is running.
2. Overrides are not applied to the files listed in the FILE parameter.
3. If FILE(*ALL) is specified, a journal name must be specified.
4. If a journal name and a list of file names are specified, all files must be currently journaled to the indicated journal.
5. This command cannot be used on or with a remote journal.
6. In multithreaded jobs, this command is not threadsafe for distributed files and fails for distributed files that use relational databases of type *SNA.

Required Parameter

FILE Specifies a maximum of 50 qualified names of the physical files for which changes are no longer being journaled.

***ALL:** All files currently being journaled to the indicated journal are to stop having their changes journaled.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the database file for which journaling has ended.

Optional Parameter

JRN Specifies the qualified name of the journal to which changes in the files are currently being journaled.

***FILE:** The journal is determined by the system from the specified file names.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

journal-name: Specify the name of the journal to which the indicated files are currently being journaled.

Example for ENDJRNPF

```
ENDJRNPF   FILE(MYLIB/MYFILE)
```

This command stops the journaling of all changes to all members of file MYFILE in library MYLIB. Changes made after this command is run are not journaled.

Error messages for ENDJRNPF

***ESCAPE Messages**

CPF6970
Access paths built over file &1 are being journaled.

CPF7002
File &1 in library &2 not a physical file.

CPF703B
Implicit end of access path journaling failed.

CPF703C
DDL transaction prevents journaling operation.

CPF703D
DDL transaction prevents journaling operation.

CPF703E
DDL transaction prevents journaling operation.

CPF7031
Cannot allocate member &3 file &1 in &2.

CPF7032
ENDJRNPF or ENDJRNAP command failed.

CPF7033
Start or end journaling failed for member &3.

CPF7034
Logical damage of file &1 in &2.

CPF704C
Journaling ended locally but distributed requests failed.

CPF704D
ENDJRNPF command failed.

CPF705A
Operation failed due to remote journal.

CPF708D
Journal receiver found logically damaged.

CPF9801
Object &2 in library &3 not found.

CPF9802
Not authorized to object &2 in &3.

CPF9803
Cannot allocate object &2 in library &3.

CPF9810
Library &1 not found.

CPF9812
File &1 in library &2 not found.

CPF9820
Not authorized to use library &1.

CPF9822
Not authorized to file &1 in library &2.

CPF9825
Not authorized to device &1.

CPF9830
Cannot assign library &1.

CPF9873
ASP status is preventing access to object.

CPF9875
Resources exceeded on ASP &1.

ENDLINRCY (End Line Recovery) Command Description

ENDLINRCY Command syntax diagram

Purpose

The End Line Recovery (ENDLINRCY) command ends error recovery procedures for a specific line. If any type of failure occurs after this command is run, an inquiry message is sent.

Use the Resume Line Recovery (RSMLINRCY) command to reestablish error recovery procedures for the line.

Required Parameter

LINE Specifies the name of the line whose recovery is stopped.

Example for ENDLINRCY

```
ENDLINRCY  LINE(NYC2)
```

This command ends error recovery procedures for the line named NYC2.

Error messages for ENDLINRCY

***ESCAPE Messages**

CPF2704
Line description &1 not found.

CPF5917
Not authorized to line description &1.

CPF5932
Cannot access line &1.

CPF5933
Line &1 not varied on.

CPF5935
Error occurred during command processing.

ENDMSF (End Mail Server Framework) Command Description

ENDMSF Command syntax diagram

Purpose

The End Mail Server Framework (ENDMSF) command ends the mail server framework jobs in the system work subsystem (QSYSWRK).

Optional Parameters

OPTION
Specifies whether the mail server framework jobs that are in the system work subsystem (QSYSWRK) end immediately or in a controlled manner.

***CNTRLD:** All mail server framework jobs are ended in a controlled manner. This allows each framework job a chance to complete processing the current mail server framework messages before it ends.

***IMMED:** All mail server framework jobs are ended immediately. Any mail server framework messages being processed at the time the job ended are processed when the mail server framework is restarted.

DELAY
Specifies the amount of time (in seconds) allowed for the mail server framework jobs to complete

their processing during a controlled end. This parameter is ignored if OPTION(*IMMED) is specified. If the jobs do not end before the end of the delay time, they are then immediately ended.

30: A maximum delay time of 30 seconds is allowed before the mail server framework jobs are ended.

delay-time: Specify the maximum amount of delay time in seconds before the jobs are ended. Valid values range from 1 through 999999.

Examples for ENDMSF

Example 1: Ending Mail Server Framework in a Controlled Manner

```
ENDMSF OPTION(*CNTRLD)  DELAY(60)
```

This command ends the mail server framework jobs in the system work subsystem in a controlled manner and has 60 seconds to complete processing any mail server framework messages currently being handled.

Example 2: Ending Mail Server Framework Immediately

```
ENDMSF OPTION(*IMMED)
```

This command ends the mail server framework jobs in the system work subsystem immediately. The mail server framework jobs do not complete processing any mail server framework messages currently being handled.

Error messages for ENDMSF

***ESCAPE Messages**

CPFAFAB
ENDMSF did not complete successfully.

CPFAFAC
ENDMSF completed successfully; however errors occurred.

CPFAFFF
Internal system error in program &1.

»

ENDMGDSYS (End Managed System Services) Command Description

Note: To use this command, you must have the 5722-MG1 (Managed System Services for iSeries) licensed program installed.

ENDMGDSYS Command syntax diagram

Purpose

The End Managed System Services (ENDMGDSYS) command ends the managed system jobs after all currently running activities are completed.

Restrictions:

1. You must have *JOBCTL authority to use the end command.
2. Public authority for this command is *EXCLUDE.

Optional Parameters

OPTION

Specifies that the managed system jobs have a controlled ending or are to be ended immediately.

***CNTRLD:** Specify that the managed system jobs end in a controlled manner.

***IMMED:** Specify that the managed system jobs end immediately. The programs that are running do not get time to perform cleanup. This option may cause undesirable results if data has been partially updated. Therefore, this option should be used only if a controlled end was unsuccessful.

DELAY

Specifies the delay time before ending is immediate.

***NOLIMIT:** The managed system jobs continue processing until the current activity processing is complete.

delay-time: Specify that the managed system jobs end immediately after the delay time.

Examples for ENDMGDSYS

Example 1: Ending Managed System Jobs

```
ENDMGDSYS
```

This command ends the managed system jobs after all currently running activities are completed.

Example 2: Ending Managed System Jobs Immediately

```
ENDMGDSYS   OPTION(*IMMED)
```

This command ends the managed system jobs immediately.

Error messages for ENDMGDSYS

*ESCAPE Messages

CPF3CD9

Requested function cannot be performed at this time.

CPF3CDA

Registration facility repository not available for use.

CPF81xx

Damaged object error messages.

CPF90FF

*JOBCTL special authority required to do requested operation.

CPF9872

Program or service program &1 in library &2 ended. Reason code &3.

MSS0059

Internal object not found or damaged.

MSS005B

Storage limit exceeded.

MSS0064

Internal object not found or damaged.

MSS0066

Internal processing error occurred.

MSS0067

Not able to allocate internal object.

MSS0321
Managed System Services/400 not active.

«

Printed in U.S.A.