



@server

iSeries

CL Commands Volume 14







@server

iSeries

CL Commands Volume 14



---

# Contents

<b>Command Descriptions</b> . . . . .	1
ENDMGRSRV (End Manager Services) Command Description . . . . .	1
ENDMOD (End Mode) Command Description . . . . .	2
ENDNFSSVR (End Network File System Server) Command Description . . . . .	3
ENDNWIRCY (End Network Interface Recovery) Command Description . . . . .	5
ENDPASTHR (End Pass-Through) Command Description . . . . .	5
ENDPEX (End Performance Explorer) Command Description . . . . .	6
ENDPFRTRC (End Performance Trace) Command Description . . . . .	10
ENDTCPPTP (End Point-to-Point TCP/IP) Command Description . . . . .	11
ENDPJ (End Prestart Jobs) Command Description . . . . .	13
ENDPRTEML (End Printer Emulation) Command Description . . . . .	15
ENDPGM (End Program) Command Description . . . . .	16
ENDPGMPRF (End Program Profiling) Command Description . . . . .	17
ENDRDR (End Reader) Command Description . . . . .	17
ENDRCV (End Receive) Command Description . . . . .	18
ENDRMTSPT (End Remote Support) Command Description . . . . .	19
ENDRQS (End Request) Command Description . . . . .	20
ENDRPCBIND (End RPC Binder Daemon) Command Description . . . . .	21
ENDSRVJOB (End Service Job) Command Description . . . . .	22
ENDSBMCRQA (End Submitted Change Request Activity) Command Description . . . . .	22
ENDSBS (End Subsystem) Command Description . . . . .	24
ENDSYS (End System) Command Description . . . . .	27
ENDSYSMGR (End System Manager) Command Description . . . . .	29
ENDS36 (End System/36) Command Description . . . . .	30
ENDTCP (End TCP/IP) Command Description . . . . .	31
ENDTCPABN (End TCP/IP Abnormal) Command Description . . . . .	33
ENDTCPENN (End TCP/IP Connection) Command Description . . . . .	34
ENDTCPIFC (End TCP/IP Interface) Command Description . . . . .	36
Notes on Route to Interface Binding . . . . .	36
ENDTCPFSVR (End TCP/IP Server) Command Description . . . . .	37
ENDTIESSN (End Technical Information Exchange Session) Command Description . . . . .	40
ENDTRC (End Trace) Command Description . . . . .	41
ENDTRPMGR (End Trap Manager) Command Description . . . . .	43
ENDUSF (End Ultimeida System Facilities) Command Description . . . . .	43
ENDWTR (End Writer) Command Description . . . . .	44
EXTMEDIBRM (Extract Media Information) Command Description . . . . .	45
GENLICKY (Generate License Key) Command Description . . . . .	45
GOTO (Go To) Command Description . . . . .	48
GO (Go to Menu) Command Description . . . . .	49
GRTACCAUT (Grant Access Code Authority) Command Description . . . . .	51
GRTOBJAUT (Grant Object Authority) Command Description . . . . .	53
GRTUSRAUT (Grant User Authority) Command Description . . . . .	60
GRTUSRPMN (Grant User Permission) Command Description . . . . .	62
GRTWSOAUT (Grant Workstation Object Authority) Command Description . . . . .	63
HLDCMNDEV (Hold Communications Device) Command Description . . . . .	66
HLDDSTQ (Hold Distribution Queue) Command Description . . . . .	68
HLDJOB (Hold Job) Command Description . . . . .	69
HLDJOBQ (Hold Job Queue) Command Description . . . . .	72
HLDJOBSCDE (Hold Job Schedule Entry) Command Description . . . . .	73
HLDJOBS (Hold Job Using Job Scheduler) Command Description . . . . .	75
HLDOUTQ (Hold Output Queue) Command Description . . . . .	75
HLDPTF (Hold Program Temporary Fix) Command Description . . . . .	77
HLDRDR (Hold Reader) Command Description . . . . .	77

HLDSPLF (Hold Spooled File) Command Description . . . . .	79
HLDSBMCRQA (Hold Submitted Change Request Activity) Command Description . . . . .	82
HLDWTR (Hold Writer) Command Description . . . . .	84
IF (If) Command Description . . . . .	85
INZBRM (Initialize BRMS) Command Description . . . . .	88
INZPCS (Initialize Client Access/400) Command Description . . . . .	89
INZDKT (Initialize Diskette) Command Description . . . . .	93
INZDSTQ (Initialize Distribution Queue) Command Description . . . . .	97
INZMEDBRM (Initialize Media using BRM) Command Description . . . . .	99
INZOPT (Initialize Optical) Command Description . . . . .	100
INZPFM (Initialize Physical File Member) Command Description . . . . .	104
INZSYS (Initialize System) Command Description . . . . .	107
INZTAP (Initialize Tape) Command Description . . . . .	108
INSPTF (Install Program Temporary Fix) Command Description . . . . .	115
INSRMTPRD (Install Remote Product) Command Description . . . . .	119
LNKDTADFN (Link Data Definition) Command Description . . . . .	122
LODRUN (Load and Run Media Program) Command Description . . . . .	124
LODIMGCLG (Load or Unload Image Catalog) Command Description . . . . .	128
LODPTF (Load Program Temporary Fix) Command Description . . . . .	129
LODQSTDB (Load Question-and-Answer Database) Command Description . . . . .	134
MRGMSGCLG (Merge Message Catalog) Command Description . . . . .	135
MRGMSGF (Merge Message File) Command Description . . . . .	136

---

# Command Descriptions



---

## ENDMGRSRV (End Manager Services) Command Description

**Note:** To use this command, you must have the 5722-MG1 (Managed System Services for iSeries) licensed program installed.

ENDMGRSRV Command syntax diagram

### Purpose

The End Manager Services (ENDMGRSRV) command ends the ability to send remote commands to managed systems and ends the ability to gather topology information from nodes and clients.

### Restrictions:

1. You must have \*JOBCTL authority to use the end command.
2. Public authority for this command is \*EXCLUDE.

### Optional Parameters

#### SERVICE

Specifies the service to be ended. One or more values may be entered.

**\*ALL:** Specify to end all the manager services.

**\*TOPOLOGY:** Specify to end ability to gather topology information from nodes and clients.

**\*RMTCMD:** Specify to end ability to send remote commands to managed systems.

#### OPTION

Specifies whether the services are to be ended in a controlled manner or immediately.

**\*CNTRLD:** Specify to end the services in a controlled manner.

**\*IMMED:** Specify to end the services immediately.

#### DELAY

Specifies the delay time in seconds to wait before ending the services immediately.

**\*NOLIMIT:** Specify to continue processing until the current activity processing is completed.

*delay-time:* Specify to end the services immediately after the specified delay time. Valid entries range from 1 to 999999.

### Examples for ENDMGRSRV

#### Example 1: Ending Central Site System Services

```
ENDMGRSRV SERVICE(*ALL)
```

This command ends the central site system services.

#### Example 2: Ending the Topology Information Services Job

```
ENDMGRSRV SERVICE(*TOPOLOGY) OPTION(*IMMED)
```

This command ends the ability to gather topology information from nodes and clients.

## Error messages for ENDMGRSRV

### \*ESCAPE Messages

#### MSS0601

\*JOBCTL special authority required for requested operation.

#### MSS0730

Error found on &1 command.



---

## ENDMOD (End Mode) Command Description

ENDMOD Command syntax diagram

### Purpose

The End Mode (ENDMOD) command ends (deactivates) a single mode or all active modes for a specific advanced program-to-program communications (APPC) remote location. The mode remains inactive until a Start Mode (STRMOD) command is run to start the mode. This command can be used to end all the sessions for a particular remote location and to cause an active switched connection to disconnect. The user can also specify how activities that have been requested on the remote system but have not yet been performed are to be handled.

The APPC, APPN and HPR topic in the Information Center has more information on the ENDMOD command.

**Restriction:** This command cannot be used to end (deactivate) Client Access/400 mode (QPCSUPP) at a remote location.

### Required Parameter

#### RMTLOCNAME

Specifies the remote location name for one or more modes which are being ended.

### Optional Parameters

**DEV** Specifies the device description name used with the remote location.

**\*LOC:** The device associated with the remote location is used. If several devices are associated with the remote location, the system determines which device is used.

*device-name:* Specify the name of the device for which one or more modes are being ended.

**MODE** Specifies the mode that is ended.

**\*NETATR:** The mode name specified in the network attributes is used.

**\*ALL:** All modes currently active by the remote location are ended.

**BLANK:** The mode name consisting of 8 blank characters is used.

*mode-name:* Specify a mode name for the device.

#### LCLLOCNAME

Specifies the local location name.

**\*LOC:** The device associated with the remote location is used. If several devices are associated with the remote location, the system determines which device is used.

**\*NETATR:** The LCLLOCNAME value specified in the system network attributes is used.



*local-location-name*: Specify the local location name.

### **RMTNETID**

Specifies the remote network ID used with the remote location.

**\*LOC**: The remote network identifier (ID) associated with the remote location is used. If several remote network IDs are associated with the remote location, the system determines which remote network ID is used.

**\*NETATR**: The LCLLOCNAME value specified in the system network attributes is used.

**\*NONE**: No remote network identifier (ID) is used.

*remote-network-ID*: Specify the name of the remote network ID used.

### **CPLPNDRQS**

Specifies how the remote location processes all pending user work before ending the mode.

**\*NO**: Requested activities currently in progress at the remote location can complete; activities that have been requested, but not started at the remote location will not be performed.

**\*YES**: All requested activities are allowed to complete before the mode is ended.

### **Example for ENDMOD**

```
ENDMOD RMTLOCNAME(APPCRLOC) MODE(APPCMOD)
```

This command ends a mode named APPCMOD for remote location APPCRLOC.

### **Error messages for ENDMOD**

#### **\*ESCAPE Messages**

#### **CPF598B**

The &1 command failed for one or more modes.

---

## **ENDNFSSVR (End Network File System Server) Command Description**

ENDNFSSVR Command syntax diagram

### **Purpose**


The End Network File System Server (ENDNFSSVR) command ends one or all of the Network File System (NFS) server daemons. For more information about these daemon jobs, see the OS/400 Network

File System Support  book.

You should use SERVER(\*ALL), which will end the daemons in the following order. (This order is the recommended order for ending the Network File System daemons.)

- The network lock manager (NLM) daemon
- The network status monitor (NSM) daemon
- The mount (MNT) daemon
- The server (SVR) daemon
- The block I/O (BIO) daemon
- The Remote Procedure Call (RPC) binder daemon

If you are choosing to end just one daemon, be sure you understand the appropriate order for ending NFS daemons and the possible consequences of ending daemons in an order other than that specified above.

For more information about ending NFS daemons, see the OS/400 Network File System Support  book.

If you attempt to end a daemon or daemons that are not running, they will not cause the command to fail, and it will continue to end other daemons you have requested to end.

To determine if an NFS daemon is running, use the Work with Active Jobs (WRKACTJOB) command and look in the subsystem QSYSWRK for existence of the following jobs:

QNFSRPCD	The RPC binder daemon
QNFSBIOD	The block I/O (BIO) daemon
QNFSNFSD	The server (SVR) daemon
QNFSMNTD	The mount (MNT) daemon
QNFSNSMD	The network status monitor (NSM) daemon
QNFSNLMD	The network lock manager (NLM) daemon

**Restriction:** You must have \*IOSYSCFG special authority to use this command.

### Required Parameter

#### SERVER

Specifies the Network File System (NFS) daemon jobs to be ended.

**\*ALL:** All NFS daemons are ended.

**\*RPC:** The NFS Remote Procedure Call (RPC) binder daemon is ended.

**\*BIO:** All NFS block I/O daemons that are running are ended.

**\*SVR:** All NFS server daemons that are running are ended.

**\*MNT:** The NFS mount daemon is ended.

**\*NSM:** The NFS network status monitor daemon is ended.

**\*NLM:** The NFS network lock manager daemon is ended.

### Optional Parameter

#### ENDJOBTIMO

Specifies the number of seconds to wait for each daemon to successfully end. If a daemon has not ended within the timeout value, the command will fail.

**30:** Wait 30 seconds for the daemon job to end.

**\*NOMAX:** Wait forever for daemons to end; do not timeout.

*timeout-value:* Specify the number of seconds to wait for each daemon to end before timing out and failing the command. Valid values range from 1 to 3600 seconds.

### Examples for ENDNFSSVR

#### Example 1: End All Daemons

```
ENDNFSSVR SERVER(*ALL)
```

This command ends all NFS daemon jobs that are running.

#### Example 2: End a Single Daemon

```
ENDNFSSVR SERVER(*MNT) ENDJOBTIMO(*NOMAX)
```

This command ends the NFS mount daemon, and waits forever for it to end. The mount daemon was previously running, and other daemons have been ended in the appropriate order.

### Error messages for ENDNFSSVR

---

## ENDNWIRCY (End Network Interface Recovery) Command Description

ENDNWIRCY Command syntax diagram

### Purpose

The End Network Interface Recovery (ENDNWIRCY) command ends automatic error recovery procedures for a network interface description.

### Required Parameter

**NWID** Specifies the name of the network interface description for which recovery is to be ended.

### Example for ENDNWIRCY

```
ENDNWIRCY NWID(ISDNNET)
```

This command ends automatic error recovery procedures for the network interface named ISDNNET.

### Error messages for ENDNWIRCY

#### \*ESCAPE Messages

##### CPF591A

Not authorized to network interface description &1.

##### CPF593A

Network interface &1 not varied on.

##### CPF593B

Network interface description &1 not found.

##### CPF593C

Cannot access network interface &1.

---

## ENDPASTHR (End Pass-Through) Command Description

ENDPASTHR Command syntax diagram

### Purpose


The End Pass-Through (ENDPASTHR) command ends a pass-through session. The ENDPASTHR command signs you off the target system, and ends the advanced program-to-program communications (APPC) session. This releases the virtual display device from the subsystem and returns it to the vary-on pending condition. The job at each intermediate node for the pass-through session also ends. Control returns to the source system for the next command following the Start Pass-Through (STRPASTHR) command.

### Note:

The ENDPASTHR command uses the SIGNOFF command as part of its processing. If the system has a SIGNOFF command that appears in the library list before QSYS/SIGNOFF, the SIGNOFF command is used by ENDPASTHR. The SIGNOFF command should not use the ENDPASTHR command. It sends the system into a loop when you end your pass-through session.

The ENDPASTHR command does not end the passthrough session when there is a secondary interactive job at the target system. One of the jobs must be ended (by using SIGNOFF or ENDJOB) before the ENDPASTHR command can be entered.

If the ENDPASTHR command is entered and there is not a pass-through session, an error message is sent.

More information about pass-through is in the Remote Work Station Support  book.

### Optional Parameter

**LOG** Specifies whether the JOBLOG is saved at the remote system.

**\*NOLIST:** The information in the job log is deleted when the job ends.

**\*LIST:** The JOBLOG is saved at the remote system.

### Example for ENDPASTHR

```
ENDPASTHR LOG(*LIST)
```

This command ends a pass-through session and prints a job log.

### Error messages for ENDPASTHR

#### \*ESCAPE Messages

##### CPF8914

ENDPASTHR command not allowed.

##### CPF8915

ENDPASTHR not allowed. System request job active.

---

## ENDPEX (End Performance Explorer) Command Description

ENDPEX Command syntax diagram

### Purpose

The End Performance Explorer (ENDPEX) command instructs the performance explorer tool to stop collecting data. The command expects a session name to accompany the request which identifies which instance of the performance explorer session to end.

The user can either end the data collection session or suspend the data collection session. ➤ If the user chooses to end the session, the collected data is put into an object of type \*MGTCOL or into a set of data base files, or it is deleted, based on the value specified for the DTAOPT parameter. ⏪

If the user chooses to suspend the collection of performance data, the session remains active. To resume data collection for a suspended session, the user can specify OPTION(\*RESUME) on a subsequent call of the STRPEX (Start Performance Explorer) command.

### ➤ Restrictions:

1. This command is shipped with PUBLIC \*EXCLUDE authority.
2. The user must have add and execute authority to the specified DTALIB and MGTCOL libraries.

3. The user must have \*OBJMGMT, \*OBJEXIST, and read authorities to the management collection object if replacing an existing management collection object.
4. 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.
5. The following user profiles have private authorities to use the command:
  - QPGMR
  - QSRV



## Optional Parameters

### SSNID

Specifies which performance explorer session to end. This is the session identifier that was specified on the STRPEX (Start Performance Explorer) command.

**\*SELECT:** A list panel of all active performance explorer data collection sessions will be displayed with an option to select which session to end. \*SELECT is only valid if the ENDPEX command is being run interactively. If the command is being run in batch, a session identifier must be specified.

*session-identifier:* Specify the performance explorer data collection session to end.

### OPTION

Specifies whether to end the data collection session or just suspend collection of performance data for the session.

**\*END:** The performance explorer session is ended. The user is prompted for a choice of three methods to handle the collected data:

1. Save the collected data to a set of database files.
2. Save the data to a single management collection object.
3. Discard the data.

**\*SUSPEND:** The performance explorer session is suspended, and the session remains active but no additional data is collected for this session. Once a session is suspended, the user can either use STRPEX with OPTION(\*RESUME) to resume data collection, end the suspended session by specifying ENDPEX with OPTION(\*END), or stop the suspended session by specifying ENDPEX with OPTION(\*STOP).

**\*STOP:** The performance explorer session is ended and the jobs are removed from the collection. The session cannot be started up again. Addresses are not resolved to object names, and no database files are created. The address data and database files can be created at a later time with the OPTION(\*END) and DTAOPT(\*LIB or \*MGTCOL) options of ENDPEX. However, performance explorer may not be able to resolve some of the addresses if objects get deleted. The longer the time between \*STOP and \*END, the greater the chance the resolved address data will be incomplete.

### DTAOPT

Specifies how to handle the collected data. The collected data can be stored in a set of database files or a management collection object (\*MGTCOL). The temporary management collection object used to hold the collected data will be deleted. To delete the temporary management collection object without storing the collected data, specify \*DLT.

**Note:** This parameter is valid only if OPTION(\*END) is specified.

**\*LIB:** Indicates to store all of the collected performance data for the session into a set of database files located in the library specified on the DTALIB parameter. The performance explorer tool creates all the necessary files if this is the first time that a library is being used to store performance data. The member name for each of the files where the session data is stored can be controlled through the DTAMBR parameter, but defaults to be the same name as the session identifier.

➤ **\*MGTCOL:** Indicates to store all of the collected data in a management collection object (type \*MGTCOL). No database files will be created. This option can be used if the data is to be shipped to another system or to your service provider for analysis. ⏪

### Single Value

**\*DLT:** The collected performance data for the session is to be deleted from the system.

### DTALIB

Specifies the name of the library that contains the set of database files where the collected performance data is stored.

**Note:** This parameter is valid only if the user specified DTAOPT(\*LIB).

**QPEXDATA:** The QPEXDATA library is the recommended library for storing data collected with the performance explorer tool. The first time the performance explorer tool is used, this library is created for the user, and a set of database files to store the information is created in that library.

*library-name:* Specifies the name of the library in which to store the collected data. If the library does not exist, the command will terminate in an error condition. After the library is created, retry the command. If the library specified does not already have the performance explorer database files, they are created and the data is stored.

### DTAMBR

Specifies the name to be used for the database file members where the collected performance data is stored. If a member does not exist by the specified name, it is created.

**Note:** This parameter is valid only when DTAOPT(\*LIB) is specified.

**\*SSNID:** The member name is the same as the value specified for the SSNID parameter.

*member-name:* Specify the member name to use when storing the collected data in performance explorer database files.

### MGTCOL

➤ Specifies the name of a management collection object to store the collected performance data.

**Note:**

This parameter is valid only if DTAOPT(\*MGTCOL) is specified.

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

**QPEXDATA:** The QPEXDATA library is the recommended library for storing data collected by the performance explorer tool. The first time the performance explorer tool is used, this library is created for the user.

*data-library-name:* Specify the name of the library to store the collected data. If the library does not exist, the command will terminate in an error condition. After the library is created, retry the command.

**\*SSNID:** The name specified for the SSNID parameter is used when creating the management collection object to contain the collected performance data.

*management-collection-name:* Specify the name to use when creating the management collection object to contain the collected performance data. <<

**RPLDTA**

>> Specifies whether to replace the data in an existing file member or management collection object with the new performance data. If DTAMBR was specified and a member with the same name already exists in any of the performance explorer database files in the specified library (DTALIB parameter), this parameter controls whether the member data is replaced. If MGTCOL was specified and an object already exists with the same name, this parameter controls whether the data in that object is replaced. <<

**\*NO:** If a member already exists with the same name, an error message is sent to the user. This prevents the user from inadvertently writing over existing data.

**\*YES:** If a member already exists with the same name, the old data is lost and is replaced by the new data.

**NBRTHD**

>> Specifies the number of concurrent threads that ENDPEX will use to process the data in the session being ended. Specifying a number greater than 1 will allow ENDPEX to take advantage of available CPU cycles, especially on a multi-processor system. While this may speed up ENDPEX processing, it may also degrade the performance of other jobs on the system. You can minimize this impact by changing the priority of the job that runs ENDPEX to a higher number. You should also verify the disk subsystem can handle the additional threads.

**\*CALC:** The system will calculate a reasonable number of threads to do the ENDPEX processing which does not use excessive CPU or disk resources.

**\*MAX:** The system will calculate a maximum number of threads to do the ENDPEX processing. An attempt will be made to maximize utilization on all resources in order to minimize ENDPEX processing time. This may cause severe degradation for all other jobs on the system.

*number-of-threads:* Specify the number of threads for ENDPEX to use to process the collected data. <<

**TEXT** Specifies the text that briefly describes the type of data collected. 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.

## Examples for ENDPEX

### Example 1: End a Session and Save the Database Files

```
ENDPEX  SSNID(TEST3) OPTION(*END)
        DTAOPT(*LIB) DTAMBR(SYS1DATA)
```

This command ends the performance explorer session named TEST3 and saves the data in a set of database files in library QPEXDATA. The member name to be used for each file is SYS1DATA.

### Example 2: End a Session and Delete the Data

```
ENDPEX  SSNID(TESTRUN) OPTION(*END) DTAOPT(*DLT)
```

This command ends the performance explorer session named TESTRUN and deletes the collected performance data.

### ▶▶ Example 3: End a Session and Save the \*MGTCOL

```
ENDPEX  SSNID(TEST3) OPTION(*END)
        DTAOPT(*MGTCOL) MGTCOL(MYLIB/SYS1DATA)
        NBRTHD(*CALC)
```

This command ends the performance explorer session named TEST3 and saves the data in a management collection object in library MYLIB in the management collection object named SYS1DATA. ENDPEX will calculate a number of threads to process this request. This number of threads will do the ENDPEX processing as quickly as possible without disrupting the rest of the system. ◀◀

## Error messages for ENDPEX

None

---

## ENDPFRTTC (End Performance Trace) Command Description

ENDPFRTTC Command syntax diagram

### Purpose

The End Performance Trace (ENDPFRTTC) command will stop the collection of performance trace data in the QPM\_STRPFRTTC trace table and optionally write performance trace data to a data base file. The QPM\_STRPFRTTC trace table will be deleted whether or not the data is written to a data base file.

This command is intended to be used to end a performance trace started via the Start Performance Trace (STRPFRTTC) command. However, it will end and try to process any active trace in the QPM\_STRPFRTTC trace table.

### Restrictions:

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

### DMPTRC

Specifies whether the trace data is dumped to the performance database file QAPMDMPT. If the data is not dumped, it will be lost when the trace table is deleted.

**\*YES:** The trace data, if any, is dumped.



**\*NO:** The trace data is not dumped.

**MBR** Specifies the member name within the QAPMDMPT database file where the trace table data is dumped.

**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.

**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 ENDPFRTRC

#### Example 1: Ending Performance Trace

```
ENDPFRTRC DMPTRC(*YES) MBR(MYDATA)
```

In this example, the current trace is ended, the data is written to file QPFRDATA/QAPMDMPT member MYDATA and the trace table is deleted, releasing the storage used by the trace.

### Error messages for ENDPFRTRC

#### \*ESCAPE Messages

Refer to the TRCINT and DMPTRC commands for messages.

---

## ENDTCPPTP (End Point-to-Point TCP/IP) Command Description

ENDTCPPTP Command syntax diagram

### Purpose

The End Point-to-Point TCP/IP (ENDTCPPTP) command is used to end a point-to-point TCP/IP session job. A session job operates in one of two possible modes. Answer mode sessions (\*ANS) allow a remote system to contact this iSeries 400 and establish a point-to-point TCP/IP session. Dial mode sessions (\*DIAL) allow this iSeries 400 to contact a remote system that supports point-to-point TCP/IP.

The TCP/IP point-to-point session jobs run in the QSYSWRK subsystem.

### Required Parameter

#### CFGPRF

Specifies which point-to-point TCP/IP sessions job or jobs should be ended.

**\*ALL:** All currently active point-to-point TCP/IP sessions jobs operating in the mode specified by the OPRMODE parameter are ended.

*generic\*-configuration-profile-name:* Specify the generic name of the point-to-point TCP/IP configuration profile. A generic name is a character string of one or more characters followed by an asterisk (\*); for example, ABC\*. If a generic name is specified, then all profiles with names that begin with the generic name are ended. If an asterisk is not included, the name is assumed to be a complete point-to-point TCP/IP configuration profile name. All currently active point-to-point TCP/IP session jobs using the profiles indicated and operating in the mode specified by the OPRMODE parameter are ended.

*configuration-profile-name:* Specify the name of a TCP/IP point-to-point configuration profile. The active point-to-point session job using this profile is ended.

## Optional Parameter

### OPRMODE

Specifies the operating mode of the TCP/IP point-to-point session job to be ended. This value must match the value specified in the point-to-point configuration profile specified by CFGPRF.

**\*ANY:** Any point-to-point TCP/IP session job that matches the configuration profile name specified on the CFGPRF parameter is ended, regardless of operating mode.

**\*ANS:** The operating mode of the session to be ended is \*ANS. All \*ANS point-to-point TCP/IP session jobs that are currently active that match the specified CFGPRF parameter will be ended.

**\*DIAL:** The operating mode of the session to be ended is \*DIAL. All \*DIAL point-to-point TCP/IP session jobs that are currently active that match the specified CFGPRF parameter will be ended.

## Examples for ENDTCPPTP

### Example 1: End a TCP/IP point-to-point session job.

```
ENDTCPPTP CFGPRF(DIALPRF)
```

This command ends the point-to-point TCP/IP session job that is using configuration profile DIALPRF. The operating mode (OPRMODE) value will default to \*ANY so the operating mode is not used in deciding whether to end the session job.

### Example 2: End all answer (\*ANS) mode TCP/IP point-to-point session jobs.

```
ENDTCPPTP CFGPRF(*ALL) OPRMODE(*ANS)
```

This command ends all active or activating point-to-point answer mode (\*ANS) TCP/IP session jobs.

### Example 3: End all TCP/IP point-to-point session jobs.

```
ENDTCPPTP CFGPRF(*ALL)
```

This command ends all active or activating point-to-point TCP/IP session jobs.

### Example 4: End all TCP/IP point-to-point session jobs starting with XYZ.

```
ENDTCPPTP CFGPRF(XYZ*)
```

This command ends all active or activating point-to-point TCP/IP session jobs that have profiles that begin with XYZ.

### Example 5: End an answer mode TCP/IP point-to-point session job using a specific profile name.

```
ENDTCPPTP CFGPRF(DIALPRF) OPRMODE(*ANS)
```

This command will end the point-to-point TCP/IP session job using profile DIALPRF if this profile is defined to run in answer mode. If the profile is defined to run in dial mode then no action will be taken.

## Error messages for ENDTCPPTP

## **\*ESCAPE Messages**

### **TCP1A1F**

Cannot process request while &3/&2/&1 using &6.

### **TCP8205**

Required object &2/&1 type \*&3 not found.

### **TCP8209**

ENDTCPPTP &1 &3 for &6/&5/&4 completed. &10 of &11 sessions ended.

---

## **ENDPJ (End Prestart Jobs) Command Description**

ENDPJ Command syntax diagram

### **Purpose**

The End Prestart Jobs (ENDPJ) command ends all jobs and any associated inline data files for a prestart job entry in an active subsystem. Jobs may be waiting for a request or may already be associated with a request. Spooled files associated with the jobs being ended can also be deleted or allowed to remain on the output queue. The limit on the number of messages being written to each of the job logs can also be changed.

**Restriction:** This command is restricted to a user with job control special authority.

### **Required Parameters**

**SBS** Specifies the name of the active subsystem that contains the prestart job entry. Specify the name of the subsystem.

**PGM** Specifies the qualified name of the program for the prestart job entry.

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.

*program-name:* Specify the name of the program for the prestart job entry.

### **Optional Parameters**

#### **OPTION**

Specifies that the job is either 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).

**Note:**

If a controlled end is issued against a job that is suspended because of a Hold Job (HLDJOB) command, a system request option 1, Transfer Secondary Job (TFRSECJOB) command, or Transfer Group Job (TFRGRPJOB) command, the delay time does not start until the job is active.

**\*IMMED:** The job is ended immediately. This option may cause undesirable results (for example, if data has been partially updated) and should be used only after a controlled end has been attempted unsuccessfully.

**Note:**

Ending a prestart job with the \*IMMED option does not immediately remove the job from the system. The system starts to perform end-of-job cleanup even though the immediate option is specified. This cleanup can be very brief or last several minutes. Functions performed during end-of-job cleanup can include:

- Closing of database files
- Spooling of the job log to an output queue
- End-of-job processing of OS/400 system internal objects
- Displaying the end-of-job display (for interactive jobs)

**DELAY**

Specifies the time (in seconds) allowed for the program to complete end-of-job processing during a controlled end. This parameter is not used if OPTION(\*IMMED) is specified. If the cleanup is not complete before the end of the delay time, the job is immediately ended. (Only end-of-job processing is performed.)

**30:** Up to 30 seconds of delay time is allowed for cleanup before the job is ended.

*delay-time:* Specify the maximum delay time (in seconds) before the job is ended. Valid values range from 1 through 999999 seconds. For additional information on ending an end-of-file delay job, refer to the EOFDLY parameter in the Override Database File (OVRDBF) command description.

**SPLFILE**

Specifies whether spooled files created by the job are retained for normal processing by a writer or are deleted.

**\*NO:** The spooled files created by the job being ended are retained 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 created by the job being ended are deleted. The job log is not deleted.

**LOGLMT**

Specifies the maximum number of entries in the message queue of the job being ended that are written to the job log. This parameter can be used to limit the number of messages written to the job log printer file, QPJOBLOG, for a job that is ended. This option is particularly useful when a job is ended and its message queue contains an excessive number of entries.

If this command is used to change the message logging limit while the messages for the ended job are being written to the spooled file, and if the new limit is greater than the number written at the time the command is entered, messages continue to be written until the new limit is reached. If the new limit is less than the number of messages already written to the spooled file, a message indicating that the limit is reached is immediately put in the spooled file as the last entry, and the

rest of the messages on the queue are ignored. If the limit is set to zero before any messages are written to the spooled file, no job log is produced for the ended job.

**\*SAME:** The value does not change.

**\*NOMAX:** There is no limit on the number of messages being logged. Messages on each job message queue are written to the log of each job.

*maximum-logged-entries:* Specify the maximum number of messages being written to the job log for each job. This value is the maximum if it is entered before the job log contains that many messages. Otherwise, the limit merely stops the process of writing any more messages to the job log. If zero is specified before any messages are written to the log, no job log is produced.

## Examples for ENDPJ

### Example 1: Ending a Job Immediately

```
ENDPJ SBS(SBS1) PGM(PJLIB/PJPGM) OPTION(*IMMED)
      SPLFILE(*YES)
```

This command ends all jobs associated with prestart job entry PJPGM in subsystem SBS1 immediately. Spooled output produced by these prestart jobs is deleted and the job log is saved.

### Example 2: Delaying a Job End

```
ENDPJ SBS(SBS2) PGM(PJPGM2) OPTION(*CNTRLD)
      DELAY(50) SPLFILE(NO)
```

This command ends all the jobs associated with prestart job entry PJPGM2 in subsystem SBS2. Spooled output for these prestart jobs is saved for normal processing by the spooling writer. The jobs have 50 seconds to perform any cleanup routines, after which they are immediately ended.

## Error messages for ENDPJ

### \*ESCAPE Messages

#### CPF0922

End Prestart Jobs command not allowed now.

#### CPF1083

Prestart jobs already are ending controlled.

#### CPF1084

Prestart jobs are already ending immediately.

#### CPF1227

No authority has been granted to use command.

#### CPF1317

No response from subsystem for job &3/&2/&1.

#### CPF1351

Function check occurred in subsystem for job &3/&2/&1.

#### CPF1834

Prestart job entry for program &1 in &2 does not exist.

---

## ENDPRTEML (End Printer Emulation) Command Description

ENDPRTEML Command syntax diagram

### Purpose

The End Printer Emulation (ENDPRTEML) command ends printer emulation without ending the job. If there is another request in the job, that request is then processed.

This command closes the file to the host system, and then writes the last data received from the host system to the spooled file or printer by closing the printer file.

In some cases, the request does not take effect immediately. The request is delayed while any of the following conditions exist in the printer emulation request:

- Printing a block sent from the host system.
- Waiting for a printer error to be cleared (for example, a paper jam).
- Waiting for a reply to a PA1 or PA2 inquiry message.
- Waiting for error recovery to be done to the host system or printer device.
- The job has been held by using the HLDJOB command. When the condition is cleared, the End Printer Emulation request takes effect, and the printer emulation request ends.

### 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 was specified on 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 ENDPRTEML

```
ENDPRTEML  EMLDEV(HOSTPRT3)
```

This command ends the printer emulation request that is using the device HOSTPRT3.

### Error messages for ENDPRTEML

#### \*ESCAPE Messages

##### CPF8599

End printer emulation function not performed.

---

## ENDPGM (End Program) Command Description

ENDPGM Command syntax diagram

### Purpose

The End Program (ENDPGM) command specifies the end of a CL program. When the command is processed, it performs the same function as a RETURN command. That is, control is returned to the command immediately following the CALL command in the calling program.

The ENDPGM command is not required at the end of a CL program. If the last statement in a CL program source file is reached and no ENDPGM command is found, an ENDPGM command is assumed by the compiler.

**Restriction:** This command is valid only within a CL program.

There are no parameters for this command.

#### Example for ENDPGM

```
PGM
*
*
*
ENDPGM
```

This program is identified by a PGM command that contains no parameters and is ended by the ENDPGM command.

#### Error messages for ENDPGM

None

---

## ENDPGMPRF (End Program Profiling) Command Description

ENDPGMPRF Command syntax diagram

#### Purpose

The End Program Profiling (ENDPGMPRF) command ends collection of program profiling data for ILE programs or service programs that have been enabled to collect profiling data.

**Restriction:** This command requires \*ALLOBJ authority.

There are no parameters for this command.

#### Example for ENDPGMPRF

```
ENDPGMPRF
```

Program profile data collection is ended.

#### Error messages for ENDPGMPRF

##### \*ESCAPE Messages

##### CPF5CAA

Unexpected error occurred during program profiling.

---

## ENDRDR (End Reader) Command Description

ENDRDR Command syntax diagram

#### Purpose

The End Reader (ENDRDR) command ends the specified diskette or database reader and makes its associated input device available to the system. The reader can be stopped either immediately, without completing the current job being read, or at the end of the current job. If the reader is in a hold state when this command is issued, the reader is stopped immediately.

## Required Parameter

**RDR** Specifies the name of the diskette or database reader being ended. The reader's associated input device is made available to the system.

## Optional Parameter

### OPTION

Specifies when the ended reader stops processing.

**\*CNTRLD:** The job is ended in a controlled manner. This allows the program to perform cleanup (end-of-job processing).

**\*IMMED:** The reader stops processing in a controlled manner immediately. The job being read in is not placed on the job queue.

## Example for ENDRDR

```
ENDRDR RDR(DISKETTE)
```

This command stops the reader DISKETTE as soon as the current job is completely read in and releases that device to the system.

## Error messages for ENDRDR

### \*ESCAPE Messages

#### CPF1317

No response from subsystem for job &3/&2/&1.

#### CPF1352

Function not done. &3/&2/&1 in transition condition.

#### CPF3312

Reader &1 neither active nor on job queue.

#### CPF3330

Necessary resource not available.

#### CPF3490

Not authorized to specified reader.

---

## ENDRCV (End Receive) Command Description

ENDRCV syntax diagram

### Purpose

The End Receive (ENDRCV) command is used to end (cancel) a request for input made by a previously issued Receive File (RCVF) or Send/Receive File (SNDRCVF) command that had WAIT(\*NO) specified. The ENDRCV command ends an input request even if the user enters the requested data at the display station at the same time that the command is processed. If the requested data is entered and is being sent to the program when the end receive operation is performed, the entered data is lost. If there is no outstanding input request, the command is ignored.

**Restriction:** This command is valid only for display files within CL programs. It cannot be used for database files.

### Optional Parameter

**DEV** Specifies the name of the display device for which the request for input is being ended.



**\*FILE:** The name of the device having the response from it ended is contained in the device file that was declared in the FILE parameter of the Declare File (DCLF) command. If the device file has more than one device name specified in it, \*FILE cannot be specified.

*device-name:* Specify the name of the display device from which a response is being ended.

### Example for ENDRCV

```
ENDRCV  DEV(MYDISPLAY)
```

Assume that a RCVF command with WAIT(\*NO) was issued earlier in the CL program to request input from the device file declared earlier in the DCLF command and from the display device MYDISPLAY. When this ENDRCV command is processed, that request for input from MYDISPLAY is ended.

### Error messages for ENDRCV

#### \*ESCAPE Messages

##### CPF0883

\*FILE not valid in DEV parameter for file &1.

##### CPF4101

File &2 in library &3 not found or inline data file missing.

---

## ENDRMTSPT (End Remote Support) Command Description

ENDRMTSPT Command syntax diagram

### Purpose

The End Remote Support (ENDRMTSPT) command varies off and deletes the configuration objects created by the Start Remote Support (STRRMTSPT) command. This command optionally deletes the QTILIB library created by the (STRRMTSPT) command.

### Restriction:

This command is not valid when the user is signed on the remote support work station.

### Optional Parameters

#### DLTLIB

Specifies whether the remote support library QTILIB is deleted when remote support is ended.

**\*NO:** The remote support library is not deleted.

**\*YES:** The remote support library is deleted.

#### » OPTION

Specifies how the remote support connection is ended.

**\*CNTRL:** The remote support connection ends when the connection timeout is reached.

**\*IMMED:** The remote support connection ends immediately. <<

### Example for ENDRMTSPT »

```
ENDRMTSPT  DLTLIB(*NO) OPTION(*IMMED)
```

This immediately ends the remote support connection and deletes the configuration objects that have been created. <<

## Error messages for ENDRMTSPT

None

---

## ENDRQS (End Request) Command Description

ENDRQS Command syntax diagram

### Purpose

The End Request (ENDRQS) command ends (cancels) a previously requested operation (command). One common use of the ENDRQS command is to cancel a request that is currently stopped at a breakpoint. This command function is also available as an option on the System Request menu.

If the ENDRQS command cannot be processed immediately because a system function that cannot be interrupted is currently running, the command is delayed until interruption is allowed.

When a request is ended, an escape message is sent to the request processing program that is currently called at the request level being canceled. Request processing programs can monitor for the escape message so that cleanup processing can be done when the request is canceled. The static storage and open files are reclaimed for any program that was called by the request processing program. None of the programs called by the request processing program are notified of the cancelation, so they have no opportunity to stop processing. To become a request processing program, the program must receive a request message.

If the ENDRQS command is in a program, that program must become a request processor before it issues this command.

More information on how to set up a program to become a request processor is in the CL Programming



book.

### Note:

External objects that are locked by the Allocate Object (ALCOBJ) command are not unlocked (deallocated) by the canceled request.

### Optional Parameter

#### RQSLVL

Specifies the (command) request level at which the command being canceled was entered.

**\*PRV:** The command entered at the immediately previous level is being canceled.

*request-level:* Specify the request level at which the command being canceled was entered. All request levels from the level specified to the current level are canceled.

### Examples for ENDRQS

#### Example 1: Ending a Command

```
CALL PRGA      (This is level 1)
  *
  *
  *
```

Breakpoint occurs

```
CALL PRGB      (This is level 2)
  *
  *
```

```
*
Breakpoint occurs
ENDRQS          (This is level 3)
```

In this example, because RQSLVL(\*PRV) is the default, the request made at level 2 is canceled. The user can then enter another command at level 2 or press F3 to show the PROGA breakpoint display again.

### Example 2: Ending a Command

```
CALL PROGA      (This is level 1)
*
*
*
Breakpoint occurs
CALL PROGB      (This is level 2)
*
*
*
Breakpoint occurs
ENDRQS RQSLVL(1) (This is level 3)
```

In this example, the request made at the highest level (CALL PROGA) is canceled. Consequently, any requests made between level 1 and level 3 are also canceled.

### Error messages for ENDRQS

None

---

## ENDRPCBIND (End RPC Binder Daemon) Command Description

ENDRPCBIND Command syntax diagram

### Purpose

The End RPC Binder Daemon (ENDRPCBIND) command ends the Remote Procedure Call (RPC) binder daemon. The RPC binder daemon job must be running to use and run Network File System (NFS) daemons and commands and some of the TI-RPC APIs.

This command can also be issued using the following alternative command:

- ENDNFSSVR SERVER(\*RPC)

If you attempt to end this daemon and it is not running, it will not cause the command to fail.

To determine if the RPC server daemon is running, use the Work with Active Jobs (WRKACTJOB) command and look in the subsystem QSYSWRK for existence of the following job:

QNFSRPCD The RPC binder daemon

### Example for ENDRPCBIND

#### Example 1: End RPC Binder Daemon

```
ENDRPCBIND
```

This command ends the RPC binder daemon job if it is running.

### Error messages for ENDRPCBIND

None

---

## ENDSRVJOB (End Service Job) Command Description

ENDSRVJOB Command syntax diagram

### Purpose

The End Service Job (ENDSRVJOB) command ends the remote job service operation. This command stops the service operation that began when the Start Service Job (STRSRVJOB) command was entered.

### Restrictions:

1. If tracing or debugging is active in the serviced job when this command is entered, the remote service operation is *not* ended.
2. This command is shipped with public \*EXCLUDE authority.
3. The following user profiles have private authorities to use the command:
  - QPGMR
  - QSYSOPR
  - QSRV
  - QSRVBAS

There are no parameters for this command.

### Example for ENDSRVJOB

ENDSRVJOB

This command stops the service operation of the job currently being serviced.

### Error messages for ENDSRVJOB

None >>

---

## ENDSBMCRQA (End Submitted Change Request Activity) Command Description

**Note:** To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

ENDSBMCRQA Command syntax diagram

### Purpose

The End Submitted Change Request Activity (ENDSBMCRQA) command ends one or more change request activities.

**Restriction:** You must be either the submitter of the change request or have \*JOBCTL special authority.

### Required Parameter

**CRQ** Specifies the change request name and the change request sequence number for the activities that are to be ended.

#### Element 1: Change Request Name

*change-request-name:* Specify the change request name of the activities to be ended.

## Element 2: Sequence Number

*sequence-number*: Specify the change request sequence number of the activities to be ended.

### Optional Parameters

#### ACTIVITY

Specifies the name of the activity that is ended.

**\*ALL:** End all activities of the specified change request.

**\*LAST:** End the activity named \*LAST. This is not the last activity added to the change request description object. This is the last activity to be run after the change request is submitted.

*activity-name*: Specify the name of the activity to end.

#### CPNAME

Specifies the APPN control point names of the managed systems on which this activity is to be performed.

##### Element 1: Network Identifier Values

**\*ALL:** All the activities for the change request specified are ended regardless of the network identifier of the managed system on which the activity is to be performed.

**\*NETATR:** Only network ID activities that match the ones defined in the network attributes for this system are ended.

*network-identifier*: Specify a network ID. Only activities for the network ID and control point name specified are ended.

##### Element 2: Control Point Values

**\*ALL:** All the activities for the change request specified are ended regardless of the control point name of the managed system on which the activity is to be performed.

**\*NETATR:** Only activities for the control point name that matches the one defined in the network attributes for this system are ended.

*control-point-name*: Specify a control point name. Only activities for the network ID and control point name specified are ended. For NetView Distribution Management Agents, the control point name is the change control client which supports numeric characters (0-9) in the first position of control point names that are valid in other platforms.

#### OPTION

Specifies how the activity is ended.

**\*CNTRL:** The activity is ended in a controlled manner. Activities that have not started to run are ended. Activities that have already started (Started or Running) only end if they do not leave partial results. The end code returned is 30.

**\*IMMED:** The activity is to be ended as soon as possible even if partial results occur. Activities that have not yet started to run are ended. Activities that have already started (Started or Running) can have their process interrupted which can cause partial results. The end code returned is 35.

**\*FRCFail:** The activity is ended immediately. If the activity cannot be ended, it is marked as ended with an end code of 39. Other activities conditioned on this activity are evaluated to determine if they are ready to run.

#### Example for ENDSBMCRQA

```
ENDSBMCRQA CRQ(CHG001 456) ACTIVITY(*ALL) CPNAME(*ALL)
```

This command ends all of the activities for the change request named CHG001 with sequence number 456 for all the nodes in a controlled manner.

## Error messages for ENDSBMCRQA

### \*ESCAPE Messages

None <<

---

## ENDSBS (End Subsystem) Command Description

ENDSBS Command syntax diagram

### Purpose

The End Subsystem (ENDSBS) command ends the specified subsystem (or all active subsystems) and specifies what happens to active work being processed. No new jobs are started in the subsystem or subsystems after this command is run.

Interactive jobs that have been transferred to a job queue by the Transfer Job (TFRJOB) command are ended as part of ending the subsystem. If an initial program load (IPL) occurs while either a batch or interactive job is on a job queue (because of the TFRJOB command), that job is removed from the job queue during IPL and its job log is produced.

You can specify that the application programs running in the subsystem are 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.

### Restrictions:

1. If the controlling subsystem is being ended, because either its name or \*ALL is specified for the SBS parameter, this command can be entered only in an interactive job. The interactive job must be in the controlling subsystem, and the command can be entered only from a work station (associated with the interactive job) whose work station entry in the controlling subsystem description specifies AT(\*SIGNON).
2. You must have job control (\*JOBCTL) authority and object operational authority to the subsystem to use this command.

### Required Parameter

**SBS** Specifies the name of the subsystem to be ended, or it specifies that all active subsystems are to be ended.

**\*ALL:** All the subsystems that are currently active are ended. All jobs are ended except the job in which this command is entered. When this value is specified, the QSYSOPR message queue should be in break delivery mode in the job issuing the ENDSBS command.

*subsystem-name*: Specify the simple name of the subsystem to be ended.

**Note:**

If the subsystem specified is the controlling subsystem, the interactive job from which the command was issued remains active. Also, if the subsystem specified is the controlling subsystem and the job that issues this command is one of two jobs that are active at the work station (through the use of the system request key or the Transfer Secondary Job (TFRSECJOB) command), neither of the jobs is forced to end. The controlling subsystem does not end until you end one of the jobs (either by signing off in one job or by ending one job from the other).

## Optional Parameters

### OPTION

Specifies whether jobs in the subsystem are ended in a controlled manner, which means that the application programs running in the subsystem are given time to perform end-of-job processing, or are ended immediately. In either case, the system performs certain cleanup processing.

**\*CNTRL**D: The jobs are ended in a controlled manner. This allows the programs that are running to perform cleanup (end of job processing). The applications have the amount of time specified on the DELAY parameter to complete cleanup before the job is ended.

**\*IMMED**: The jobs are ended 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 \*CNTRL 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) allowed in which to complete a controlled subsystem end operation. If this amount of time is exceeded, any jobs still running in the subsystem are ended immediately.

**\*NOLIMIT**: The amount of time in which to complete a controlled end operation is not limited.

*delay-time*: Specify the number of seconds in which the end operation is completed. Valid values range from 1 through 99999 seconds.

### ENDSBSOPT

Specifies the options to take when ending the active subsystems. In general, specifying these options will improve the performance of the ENDSBS command. Each option has certain side effects that you need to analyze before using that option.

This parameter has no effect on jobs that are already in the ending status.

**\*DFT**: The subsystems will end with no special ending options.

- Joblogs will be produced.
- The run priority will not change.
- The timeslice value will not change.

**\*NOJOBLOG**: No joblogs will be created for jobs that are ended due to this command being invoked. This includes subsystem monitor jobs and all user jobs in the subsystem. This option can

significantly reduce the amount of time necessary to complete the ENDSBS command. However, if a problem occurs in a job, there will be no joblog to record the problem, which may make problem diagnosis difficult or impossible.

**\*CHGPTY:** The CPU priority of jobs that are ending is changed to a higher value (worse priority). The remaining active jobs on the system may have better performance when \*CHGPTY is specified. However, jobs that are ending may take longer to finish. This option is ignored if the subsystem is ending controlled. But if the DELAY time limit expires, this option will take affect immediately.

**\*CHGTSL:** The timeslice of jobs that are ending is changed to a lower value. The remaining active jobs on the system may have better performance when \*CHGTSL is specified. However, jobs that are ending may take longer to finish. This option is ignored if the subsystem is ending controlled. But if the DELAY time limit expires, this option will take affect immediately.

**Note:**

Specifying \*CHGPTY and \*CHGTSL will reduce the impact on other active jobs on the system, but this may cause undesirable delays if there are active workstations that were allocated to the ending subsystem. It may take longer for those workstations to have their signon screens re-displayed since the job using the display must end before the workstation is ready to be allocated to another subsystem.

**Example for ENDSBS**

```
ENDSBS SBS(QBATCH) OPTION(*CNTRLD) DELAY(60)
```

This command ends all active jobs in the QBATCH subsystem and ends the subsystem. The active jobs are allowed 60 seconds to perform application-provided end-of-job processing.

**Error messages for ENDSBS**

**\*ESCAPE Messages**

**CPF1001**

Wait time expired for system response.

**CPF1032**

System ending with \*CNTRLD option.

**CPF1033**

System ending with \*IMMED option.

**CPF1034**

All subsystems ending with \*CNTRLD option.

**CPF1035**

Subsystems ending with \*IMMED option.

**CPF1036**

System powering down with \*CNTRLD option.

**CPF1037**

System powering down with \*IMMED option.

**CPF1038**

No authority to use command.

**CPF1052**

ENDSBS \*ALL not allowed in current environment.



**CPF1053**

Ending controlling subsystem &1 not allowed.

**CPF1054**

No subsystem &1 active.

**CPF1055**

Subsystem &1 ending with \*CNTRLD option.

**CPF1056**

Subsystem &1 already ending with \*IMMED option.

**CPF1081**

Controlling subsystem already ending to a single job.

**CPF1091**

Function check occurred in system arbiter.

---

## ENDSYS (End System) Command Description

ENDSYS Command syntax diagram

### Purpose

The End System (ENDSYS) command ends most activity on the system and leaves the system in a condition in which only the console is active in the controlling subsystem. This is normally done so that the programs to diagnose equipment problems can be run. This condition is called the restricted state and is required for operations like SAVSYS or RCLSTG.

If two jobs are active in the controlling subsystem at the console (through the use of the system request key), neither of the jobs is forced to end. The ENDSYS command cannot complete running until the user ends one of the jobs (either by signing off in one job or by canceling one job from the other).

All active subsystems are notified that an end system operation is in process. No new jobs or routing steps can be accepted by the subsystems. The ENDSYS command also specifies what happens to all active work.

Interactive jobs that are transferred to a job queue by the Transfer Job (TFRJOB) command are ended as part of subsystem ending. If an initial program load (IPL) occurs while either a batch or interactive job is on a job queue (because of the TFRJOB command), that job is removed from the job queue during IPL and its job log is produced.

### Restriction:

This command can be entered only in an interactive job in the controlling subsystem. To use this command, the user must have job control (\*JOBCTL) authority. This command is not allowed in a pass-through job or in a work station function job.

### Optional Parameters

#### OPTION

Specifies whether all active jobs are ended in a controlled manner (which lets the application programs perform end of processing) or immediately. In either case, the system performs certain job cleanup functions.

**\*CNTRLD:** The job is ended in a controlled manner. This allows the program to perform cleanup (end-of-job processing).

**\*IMMED:** The job is ended immediately. This option may cause undesirable results (for example, if data has been partially updated) and should be used only after a controlled end has been attempted unsuccessfully.

## DELAY

Specifies the amount of time (in seconds) that the controlled end operation is allowed. If this amount of time is exceeded and the end operation is not complete, any jobs still being processed are ended immediately, except for those running long-running system instructions.

**\*NOLIMIT:** The amount of time to complete a controlled end operation is not limited.

*delay-time:* Specify the number of seconds in which the end operation is completed. Valid values range from 1 through 99999 seconds.

## ENDSBSOPT

Specifies the options to take when ending the active subsystems. In general, specifying these options will improve the performance of the ENDSYS command. Each option has certain side effects that you need to analyze before using that option.

This parameter has no effect on jobs that are already in the ending status.

**\*DFT:** The subsystems will end with no special ending options.

- Joblogs will be produced.
- The run priority will not change.
- The timeslice value will not change.

**\*NOJOBLOG:** No joblogs will be created for jobs that are ended due to this command being invoked. This includes subsystem monitor jobs and all user jobs in the subsystem. This option can significantly reduce the amount of time necessary to complete the ENDSYS command. However, if a problem occurs in a job, there will be no joblog to record the problem, which may make problem diagnosis difficult or impossible.

**\*CHGPTY:** The CPU priority of jobs that are ending is changed to a higher value (worse priority). The remaining active jobs on the system may have better performance when \*CHGPTY is specified. However, jobs that are ending may take longer to finish. This option is ignored if the subsystem is ending controlled. But if the DELAY time limit expires, this option will take affect immediately.

**\*CHGTSL:** The timeslice of jobs that are ending is changed to a lower value. The remaining active jobs on the system may have better performance when \*CHGTSL is specified. However, jobs that are ending may take longer to finish. This option is ignored if the subsystem is ending controlled. But if the DELAY time limit expires, this option will take affect immediately. >>

## CONFIRM

Specifies whether the request should be confirmed before the system is ended.

**\*ENVVAR:** The value in environment variable QIBM\_ENDSYS\_CONFIRM is used to determine whether the request should be confirmed. If the value is set to \*YES or \*NO, the action described below for that value is taken. If the environment variable is not defined or not set to one of these values, then there is no confirmation.

**\*YES:** A confirmation panel is displayed when the ENDSYS command is issued.

**\*NO:** There is no confirmation when the ENDSYS command is issued. <<

## Examples for ENDSYS

### Example 1: Ending System Activity

ENDSYS

This command ends the system activity after all active jobs in the system are allowed to perform their own end of processes. The amount of time the end can take is not limited.

### Example 2: Ending System Activity After Jobs are Ended

```
ENDSYS OPTION(*IMMED)
```

This command ends system activity after all active jobs are immediately ended.

### Error messages for ENDSYS

#### \*ESCAPE Messages

##### CPF1001

Wait time expired for system response.

##### CPF1017

ENDSYS not allowed when console powered or varied off.

##### CPF1032

System ending with \*CNTRLD option.

##### CPF1033

System ending with \*IMMED option.

##### CPF1034

All subsystems ending with \*CNTRLD option.

##### CPF1035

Subsystems ending with \*IMMED option.

##### CPF1036

System powering down with \*CNTRLD option.

##### CPF1037

System powering down with \*IMMED option.

##### CPF1038

No authority to use command.

##### CPF1051

Command can only be run in controlling subsystem.

##### CPF1082

Controlling subsystem already ending to single job.

##### CPF1091

Function check occurred in system arbiter.



---

## ENDSYSMGR (End System Manager) Command Description

**Note:** To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

ENDSYSMGR Command syntax diagram

### Purpose

The End System Manager (ENDSYSMGR) command ends the jobs in the QSYSWRK subsystem that run the System Manager iSeries functions.

### 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 change request manager monitor job and the electronic customer support control job have a controlled ending, or are to be ended immediately.

**\*CNTRL**D: The change request manager monitor job and the electronic customer support control job ends in a controlled manner.

**\*IMMED**: The change request manager monitor job and the electronic customer support control job ends immediately.

#### DELAY

Specifies the delay time before ending is immediate.

**\*NOLIMIT**: The change request manager monitor job and the electronic customer support control job continue processing until the current activity processing is complete. Both jobs name a controlled ending with the maximum value of 999999 seconds specified.

*delay-time*: Specify that the change request manager monitor job and the electronic customer support control job end immediately after the delay time.

### Examples for ENDSYSMGR

#### Example 1: Ending jobs

```
ENDSYSMGR
```

This command ends the jobs in the QSYSWRK subsystem that run the System Manager function in a controlled manner.

#### Example 2: Ending jobs immediately

```
ENDSYSMGR OPTION(*IMMED)
```

This command ends the jobs in the QSYSWRK subsystem that run the System Manager function in an immediate manner.

### Error messages for ENDSYSMGR

#### \*ESCAPE Messages

None <<

---

## ENDS36 (End System/36) Command Description

ENDS36 Command syntax diagram

### Purpose

The End System/36 (ENDS36) command allows the user to end the System/36 Environment session that was started with a Start System/36 (STRS36) command.

There are no parameters for this command.

### Example for ENDS36

This command immediately ends the System/36 Environment session and any programs or procedures that are running in the System/36 Environment. If the ENDS36 command is in a procedure or in a program, the statements following the command are ignored.

No error messages.

---

## ENDTCP (End TCP/IP) Command Description

ENDTCP Command syntax diagram

### Purpose

The End TCP/IP (ENDTCP) command ends TCP/IP processing.

### Attention:

There is no confirmation display shown when ENDTCP is entered. The ENDTCP command must be used carefully. When it is used, it ends all TCP/IP processing on the iSeries 400 that you are working on.

If OPTION(\*IMMED) is specified for the ENDTCP command, the following is true:

- All TCP/IP connections are ended. This affects all currently active applications using sockets or the Pascal API.
- Unless ENDSVR(\*NO) is specified, all TCP/IP server jobs for TELNET, FTP, SMTP, LPD, HTTP, WSG, POP, Routed, DCE, DIRSRV, and the SNMP agent that are currently active in the QSYSWRK subsystem are ended. See the description of the ENDSVR parameter below.
- All active TCP/IP interfaces are ended.

If OPTION(\*CNTRL) is specified for the ENDTCP command, the following is true:

- No new open operations are allowed to TCP, UDP, or raw sockets.
- A job is submitted to the QSYSWRK subsystem that will, after the time indicated in the DELAY parameter value has expired, do an ENDTCP \*IMMED operation.
- An ENDTCP OPTION(\*IMMED) can be submitted at any time after issuing ENDTCP OPTION(\*CNTRL). This cancels the controlled end. TCP/IP processing is ended immediately when the ENDTCP OPTION(\*IMMED) is issued.

### Optional Parameters

#### OPTION

Specifies whether TCP/IP processing is ended in an immediate or controlled manner.

**\*IMMED:** TCP/IP processing is ended immediately.

#### Attention:

The ENDTCP OPTION(\*IMMED) command should be used carefully. Partially updated data may result if an application is processing data and has not completed an operation when the ENDTCP \*IMMED command is issued. It is suggested that you do the following:

- Notify all users before issuing the ENDTCP command so that they can end their applications.
- Issue the ENDTCP command at a time when you know no TCP/IP traffic is occurring on the iSeries 400. To display the current TCP/IP traffic on the iSeries 400, use option 3 on the Work with TCP/IP Status (WRKTCPSTS or NETSTAT) command.

**\*CNTRL**D: TCP/IP processing is ended in a controlled manner. Applications using TCP/IP are given time to complete their processing. New application processing is not allowed. After the specified period of time elapses, the processing for ENDTCP OPTION(\*IMMED) is performed.

The controlled end processing does not do any of the following:

- It does not monitor to see if all TCP/IP processing has completed before the specified period of time has elapsed.
- It does not notify an application that is actively using a TCP/IP connection that TCP/IP processing will be ended.

## DELAY

Specifies the amount of time (in seconds) allowed in which to complete a controlled end of TCP/IP processing. After this period of time all TCP/IP processing is ended immediately.

*delay-time*: Specify the number of seconds in which the end operation is completed. Valid values range from 1 through 86400 seconds.

## ENDSVR

Specifies whether or not all TCP/IP application server jobs are ended when the End TCP/IP (ENDTCP) command ends TCP/IP processing.

**Caution:** Before specifying \*NO for this parameter, please consider the following:

- It is not possible to end all the TCP/IP processing on your system without affecting the applications which use TCP/IP.
- If TCP/IP processing is ended and no form of TCP/IP emulation (such as AnyNet) is active, then TCP/IP applications which are not restarted will not function correctly.
- Applications that use the Pascal API must always be ended and restarted whenever TCP/IP processing is ended and restarted.

**\*YES:** The ENDTCP command ends all TCP/IP application servers prior to ending TCP/IP processing.

**\*NO:** The ENDTCP command does not end any TCP/IP application server jobs when it ends TCP/IP processing.

### Note:

ENDTCP ENDSVR(\*NO) can be used to end TCP/IP processing without disturbing the operation of jobs using AnyNet. TCP/IP processing will be ended, however TCP/IP application servers that are using AnyNet will continue to function.

If both TCP/IP and AnyNet are inactive, use the End TCP/IP Server (ENDTCPSVR) command to end TCP/IP application server jobs.

## Examples for ENDTCP

### Example 1: Ending TCP/IP Immediately

```
ENDTCP *IMMED
```

This command ends all TCP/IP processing on the iSeries 400 immediately.

### Example 2: Ending TCP/IP in a Controlled Time

```
ENDTCP OPTION(*CNTRL) DELAY(120)
```

This command ends all TCP/IP processing after 120 seconds have expired. During this time, new TCP/IP processing is not allowed.

### Example 3: Ending TCP/IP Immediately Without Ending Application Servers

```
ENDTCP *IMMED ENDSVR(*NO)
```

This command ends all TCP/IP processing on the iSeries 400 immediately. However, any TCP/IP application servers (FTP, SMTP, and so on) that are active are not ended when TCP/IP processing is ended.

#### Error messages for ENDTCP

##### \*ESCAPE Messages

###### TCP1A13

Another job is starting or ending TCP/IP or IP over SNA.

###### TCP1A70

&1 not active.

###### TCP1A72

TCP/IP already ending with \*CNTRLD option.

###### TCP1A73

Internal object damaged.

###### TCP1A74

Error occurred submitting job.

###### TCP1A77

&1 completed successfully; however errors occurred.

###### TCP9999

Internal system error in program &1.

---

## ENDTCPABN (End TCP/IP Abnormal) Command Description

ENDTCPABN Command syntax diagram

### Purpose

The End TCP/IP Abnormal (ENDTCPABN) command is used to force TCP/IP processing to terminate. It may only be used after attempting to use the End TCP/IP (ENDTCP) command with OPTION(\*IMMED) specified.

The ENDTCPABN command cannot be issued until either the ENDTCP command has completed or until 10 minutes have passed following the request for TCP/IP immediate ending. This allows sufficient time for normal TCP/IP ending functions to occur.

Successful completion of ENDTCPABN processing should permit TCP/IP to be started without a system IPL. Issuing the ENDTCPABN command does not directly affect system termination. The next system end will **not** be marked as ABNORMAL as a result of ENDTCPABN processing.

### Restrictions:

1. This command is shipped with public \*EXCLUDE authority. The QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles are shipped with private authorities to use this command.
2. Users cannot run the ENDTCPABN command until ten minutes *after* immediate ending of TCP/IP is started.

### Example for ENDTCPABN

```
ENDTCPABN
```

## Error messages for ENDTCPABN

### \*ESCAPE Messages

#### TCP1A66

ENDTCPABN not allowed at this time. Reason &1.

---

## ENDTCPENN (End TCP/IP Connection) Command Description

ENDTCPENN Command syntax diagram

### Purpose

The End TCP/IP Connection (ENDTCPENN) command is used to end a Transmission Control Protocol/Internet Protocol (TCP/IP) connection. This command ends a connection immediately and should be used only when a normal end is not possible.

#### Note:

The ENDTCPENN command is usually used by specifying option 4 on the Work with TCP/IP Connection Status list of the WRKTCPSTS (NETSTAT) display. The ENDTCPENN command is provided as a separate command to give system administrators control over this function. By limiting the authority to the ENDTCPENN command, the system administrator limits which users can end TCP/IP connections without restricting access to the NETSTAT utility.

### Required Parameters

#### PROTOCOL

Specifies the protocol used by the connection that is to be ended. The protocol value must be either \*TCP or \*UDP.

**\*UDP:** The connection was created for use with the User Datagram Protocol (UDP).

**\*TCP:** The connection was created for use with the Transmission Control Protocol (TCP).

#### LCLINTNETA

Specifies the local internet address of the connection to end. This parameter is required for both UDP and TCP.

**\***: The local internet address was left unspecified when this connection was opened.

*local-internet-address*: Specify the local internet address. The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. An internet address is not valid if it has a value of all binary ones or all binary zeros for the network identifier (ID) portion or the host ID portion of the address. If the internet address is entered from a command line, the address must be enclosed in apostrophes.

#### LCLPORT

Specifies the local port number of the connection to end. This parameter is required for both UDP and TCP. A decimal number identifying a local port must always be specified for this command.

*local-port*: Specify the local port number of the connection to end. Valid values range from 1 through 65535.

#### Attention:

Ports 1 through 1024 are reserved for use by system-supplied TCP/IP applications. If the user specifies ports 1 through 1024, it can affect the operation of those applications.



## Optional Parameters

### RMTINTNETA

Specifies the remote internet address of the connection to end. This parameter is required for TCP.

\*: The remote internet address was left unspecified when this connection was opened.

*remote-internet-address*: Specify the remote internet address. The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. An internet address is not valid if it has a value of all binary ones or all binary zeros for the network identifier (ID) portion or the host ID portion of the address. If the internet address is entered from a command line, the address must be enclosed in apostrophes.

### RMTPORT

Specifies the remote port number of the connection to end.

This parameter is required for TCP.

\*: The remote port number was left unspecified when this connection was opened.

*remote-port*: Specify the remote port number of the connection to end.

## Examples for ENDTCPN

### Example 1: Ending a TCP Connection

```
ENDTCPN PROTOCOL(*TCP)
LCLINTNETA('9.5.1.109') LCLPORT(13054)
RMTINTNETA('9.130.28.144') RMTPORT(23)
```

This command ends the TCP connection between local port 13054 for local internet address 9.5.1.109 and remote port 23 for remote internet address 9.130.28.144. The TCP/IP protocol stack ends all activity on the connection and returns the resources to the free storage pools.

### Example 2: Closing a UDP Socket

```
ENDTCPN PROTOCOL(*UDP)
LCLINTNETA('9.130.28.144') LCLPORT(596)
```

This command closes the UDP socket using local port 596 and local internet address 9.130.28.144. The TCP/IP protocol stack ends all activity on the connection and returns the resources to the free storage pools.

### Example 3: Ending a LISTEN State TCP Socket

```
ENDTCPN PROTOCOL(*TCP)
LCLINTNETA(*) LCLPORT(5023)
RMTINTNETA(*) RMTPORT(*)
```

This command ends the TCP socket that is listening on local port 5023. The application that created this socket did not specify a local internet address. The socket is closed and the local port is made available for use by another application.

## Error messages for ENDTCPN

### \*ESCAPE Messages

#### TCP2670

Not able to complete request. TCP/IP services are not available.

#### TCP3B01

Not able to end TCP connection &3 &4, &5 &6.

## TCP3B02

Not able to close UDP socket &3 &4.

## TCP9999

Internal system error in program &1.

---

# ENDTCPIFC (End TCP/IP Interface) Command Description

ENDTCPIFC Command syntax diagram

## Purpose

The End TCP/IP Interface (ENDTCPIFC) command is used to end a Transmission Control Protocol/Internet Protocol (TCP/IP) interface. When an interface is ended with this command, datagrams addressed to the IP addresses associated with this interface will no longer be accepted. However, the operation of any other TCP/IP or IP over SNA interface that is using the same line description as the the interface being ended is not affected.

This command can be used to end an interface that was previously started by the Start TCP/IP Interface (STRTCPIFC) or Start TCP (STRTCP) command.

Use this command to end all TCP/IP interfaces prior to ending TCP/IP. Also, use this command to end all TCP/IP interfaces prior to varying off a device, controller, or line associated with TCP/IP. Failure to do so may cause unpredictable results.

## Notes on Route to Interface Binding

Interfaces define direct paths to networks or subnetworks that this iSeries 400 is directly attached to. Routes define indirect paths. A route defines the first hop on the path to a network or subnetwork that this iSeries 400 is not directly attached to.

Routes are bound to interfaces using a best match first algorithm. This algorithm is based on the state of the interface and on the type of service (TOS) specified for the route and interface. When ending an interface, the routes associated with the interface can move to another existing active interface. This provides the widest available level of connectivity.

## Required Parameter

### INTNETADR

Specifies the internet address of an interface that had previously been added to the TCP/IP configuration with the Add TCP/IP Interface (ADDTCPPIFC) command and which had been previously started by the STRTCPIFC or STRTCP command. The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. An internet address is not valid if it has a value of all binary ones or all binary zeros for the network identifier (ID) portion or the host ID portion of the address. If the internet address is entered from a command line, the address must be enclosed in apostrophes.

## Examples for ENDTCPIFC

### Example 1: Ending an X.25 Interface

```
ENDTCPIFC INTNETADR('9.5.11.125')
```

This command causes the TCP/IP protocol stack to deactivate (end) the interface associated with the internet address 9.5.11.125.

### Example 2: Ending a Token-Ring Interface

```
ENDTCPIFC INTNETADR('156.93.81.7')
```

This command causes the TCP/IP protocol stack to deactivate (end) the interface associated with the internet address 156.93.81.7.

### **Error messages for ENDTCPIFC**

#### **\*ESCAPE Messages**

##### **TCP1B15**

Line description &2 unusable. Internal errors encountered.

##### **TCP1B61**

Unable to determine if &1 interface ended.

##### **TCP1B62**

Cannot determine if &1 interface ended.

##### **TCP1B65**

&2 interface not ended. Reason &1.

##### **TCP1B72**

&1 interface not ended. &1 interface is not active.

##### **TCP1B73**

&1 interface not ended. &1 interface not defined in the TCP/IP configuration.

##### **TCP1B74**

&1 interface not ended. Line description &2 not found.

##### **TCP1B85**

Unable to submit request to end interface &1.

##### **TCP265F**

INTNETADR parameter value &2 not valid.

##### **TCP265F**

INTNETADR parameter value &2 not valid.

##### **TCP9999**

Internal system error in program &1.

---

## **ENDTCPSVR (End TCP/IP Server) Command Description**

ENDTCPSVR Command syntax diagram

### **Purpose**

The ENDTCPSPVR command is used to end the TCP/IP application server jobs that are specified in the SERVER parameter. If the jobs have any current active connections, these connections are ended immediately. If the ENDTCPSPVR command is used to end a server that is not active, a diagnostic message may be returned.

➤ The End TCP/IP Server command can only be used when TCP/IP is fully operational. The interface server job QTCPIP must be available. When the iSeries is in restricted state, this command is not allowed.

Additional servers can automatically be added to the list of servers that ENDTCPSPVR will support by using the ADDTCPSVR (Add TCP/IP Server) CL command. ⚡

### **Optional Parameters**

#### **SERVER**

Specifies which of the TCP/IP application server jobs are to be ended by this command.

## Single Value

\***ALL**: All of the TCP/IP server jobs are ended.

## TCP/IP Application Servers

➤ Additional servers could also be available if they were added with the ADDTCPSVR command. Prompt on the server parameter for a complete list of supported servers. ⏪

\***SNMP**: All jobs associated with the SNMP agent in the QSYSWRK subsystem are ended.

\***ROUTED**: All jobs associated with the Routed server are ended.

\***BOOTP**: All jobs associated with the BOOTP server are ended.

\***TFTP**: All jobs associated with the TFTP servers are ended.

\***DNS**: All jobs associated with the DNS server are ended.

\***DHCP**: All jobs associated with the DHCP server are ended.

\***DDM**: The job that listens for DDM and DRDA connect requests and dispatches pre-started jobs to service them is ended.

\***TELNET**: All jobs associated with the TELNET servers are ended.

\***FTP**: All jobs associated with the FTP servers are ended.

\***SMTP**: All jobs associated with the SMTP server that are running in the QSYSWRK subsystem are ended. The bridge job running in the QSNADS subsystem is not ended.

\***LPD**: All jobs associated with the LPD servers are ended.

\***HTTP**: All jobs associated with the World Wide Web HyperText Transfer Protocol (HTTP) servers are ended. ⏪

\***POP**: All jobs associated with the Post Office Protocol (POP) Version 3 servers are ended.

\***REXEC**: All jobs associated with the REXEC servers are ended.

\***NSMI**: All jobs associated with the Network Station Manager inventory (NSMI) server are ended.

\***DCE**: All jobs associated with the DCE servers are ended.

\***DIRSRV**: All jobs associated with the Directory Services (DIRSRV) server are ended.

\***USF**: All jobs associated with the Ultimedia System Facilities (USF) server are ended.

\***NSLD**: All jobs associated with the Network Station Login Daemon (NSLD) server are ended.

\***INETD**: All jobs associated with the Internet daemon (INETD) super-server are ended.

\***MGTC**: All jobs associated with the Management Central (MGTC) server are ended.

\***ONDMD**: All jobs associated with the OnDemand Client for iSeries 400 (ONDMD) server are ended.

\***NETSVR**: All jobs associated with the NetServer (NETSVR) server are ended.

\***DLFM**: All jobs associated with the DataLink File Manager (DLFM) server are ended.

\***VPN**: All jobs associated with the Virtual Private Network (VPN) server are ended.

\***EDRSQL**: All jobs associated with the Extended Dynamic Remote SQL (EDRSQL) server are ended.

\***HOD**: All jobs associated with the Host On Demand (HOD) server are ended.

\***ODPA**: All jobs associated with the On-Demand Platform Authentication (ODPA) server are ended.

- \***QOS**: All jobs associated with the Quality of Service (QoS) server are ended.
- \***NTP**: All jobs associated with the Simple Network Time Protocol (SNTP) server are ended.
- \***TCM**: All jobs associated with the Triggered Cache Manager (TCM) server are ended.
- \***DOMINO**: All jobs associated with the Domino server are ended.
- \***LQP**: All jobs associated with the Lotus QuickPlace (LQP) server are ended.
- \***WEBFACING**: All jobs associated with the 5250 WebFacing server are ended.
- » \***DBG**: All jobs associated with the Debug server are ended.
- \***ASFTOMCAT**: All jobs associated with the Apache Software Foundation (ASF) Tomcat server are ended. «

## HTTPSVR

Specifies the name of the HTTP server instance to end.

The SERVER parameter specified must be \*HTTP or this parameter will be ignored.

If multiple HTTP server instances have been defined, you can choose to end all instances, or end one specific instance by specifying the instance name to be ended.

### Single Value

\***ALL**: All server instances for the HTTP server that are currently running will be ended.

*server-instance-name*: The specified HTTP server instance will be ended.

\***ADMIN**: The administration HTTP server will be ended. The administration server is an instance of the HTTP server that allows administration of certain iSeries functions using a web browser.

## DNSSVR

Specifies the name of the DNS server instance to end.

The SERVER parameter specified must be \*DNS or this parameter will be ignored.

If multiple DNS server instances have been defined, you can choose to end all instances, or end one specific instance by specifying the instance name to be ended.

### Single Value

\***ALL**: All server instances for the DNS server that are currently running will be ended.

*server-instance-name*: The specified DNS server instance will be ended.

## TCMSVR

Specifies the name of the TCM instance to end.

The SERVER parameter specified must be \*TCM or this parameter will be ignored.

If multiple TCM instance names have been defined, you can choose to end all instances, or end one specific instance by specifying the instance name to be ended.

### Single Value

\***NONE**: No instances for the TCM server that are currently running will be ended.

*instance-name*: The specified TCM instance name will be ended.

\***ALL**: All instances for the TCM server that are currently running will be ended.

## » TOMCATSVR

Specifies the name of the Tomcat instance to end.

The SERVER parameter specified must be \*ASFTOMCAT or this parameter will be ignored.

If multiple Tomcat instance names have been defined, you can choose to end all instances, or end one specific instance by specifying the instance name to be ended.

### Single Value

**\*NONE:** No instances for the Tomcat server that are currently running will be ended.

*server-instance-name:* The specified Tomcat instance name will be ended.

**\*ALL:** All instances for the Tomcat server that are currently running will be ended. <<

## Examples for ENDTCPSPVR

### Example 1: Ending All TCP/IP Servers

```
ENDTCPSPVR SERVER(*ALL)
```

This command ends all active TCP/IP application server jobs.

### Example 2: Ending the LPD Servers

```
ENDTCPSPVR SERVER(*LPD)
```

This command ends the TCP/IP LPD application server jobs.

### Example 3: Ending a specific HTTP server instance

```
ENDTCPSPVR SERVER(*HTTP) HTTPSPVR(http1)
```

This command ends the TCP/IP HTTP application server instance named 'http1'.

### Example 4: Ending a specific DNS server instance

```
ENDTCPSPVR SERVER(*DNS) DNSSVR('dns1')
```

This command ends the TCP/IP DNS application server instance named 'dns1'.

## Error messages for ENDTCPSPVR

### \*ESCAPE Messages

#### TCP1A0A

&1 ended abnormally.

#### TCP1A0C

TCP/IP LPP must be installed or at correct version level.

#### TCP1A11

&1 failed.

---

## ENDTIESSN (End Technical Information Exchange Session) Command Description

ENDTIESSN Command syntax diagram

### Purpose

The End Technical Information Exchange Session (ENDTIESSN) command allows the user to disconnect the communications line used for TIE batch commands.

There are no parameters for this command.

### Example for ENDTIESSN

```
ENDTIESSN
```

This command ends the TIE function by disconnecting the communications line used for TIE batch commands.

### Error messages for ENDTIESSN

None

---

## ENDTRC (End Trace) Command Description

ENDTRC Command syntax diagram

### Purpose

The End Trace (ENDTRC) command ends a trace session that was started by a STRTRC (Start Trace) command.

### Restrictions:

1. To use this command you must have \*SERVICE special authority, or be authorized to the Service Trace function of the operating system 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.
2. To use this command with the DTAOPT(\*LIB) parameter you must have authority to the library and the database files within that library where the trace data is stored.
3. If PRTRC(\*YES) is specified, you must have authority to the PRTRC (Print Trace) command.

### Required Parameter

#### SSNID

Specifies a session identifier for this trace. This name must match the session identifier of a trace that had been previously started and is still active.

**\*PRV:** The trace session most recently started by the same user who is running this ENDTRC command will be ended. For example, if the job running the ENDTRC command is running under user profile BOB, the last trace session started under user profile BOB is ended.

*session-identifier:* Specify the session identifier of the trace to end.

### Optional Parameters

#### DTAOPT

Specifies whether the trace data that has been collected is stored into database files so it can be printed later or if the trace data is deleted.

**\*LIB:** The trace data will be copied into database files. The PRTRC parameter on this command or the Print Trace (PRTRC) command can be used to format and print the data.

**\*DLT:** The trace data will be deleted from the internal buffers where it was collected.

## DTALIB

Specifies the name of the library in which the trace data will be stored. A set of database files will be created in this library to contain the trace data. The files will be created if they do not already exist.

**Note:** This parameter is valid only if DTAOPT(\*LIB) is specified.

**\*CURLIB:** The trace data is stored in files in the current library for the job. If no library is specified as the current library for the job, QGPL is used.

*library-name:* Specify the library to contain the trace data files.

## RPLDTA

Specifies whether trace data that was collected by a previous trace session with the same session identifier is replaced with new trace data. This is determined by checking if the set of database files where the trace data is to be stored already have file members with the same name as the specified trace session identifier (SSNID parameter).

**Note:** This parameter is valid only if DTAOPT(\*LIB) is specified.

**\*YES:** If trace data already exists with the specified session identifier, the old trace data is lost and replaced by the new trace data.

**\*NO:** If trace data already exists for the specified session, an error message is sent to the user.

## PRTTTC

Specifies whether trace data is formatted and printed after it is stored in the trace database files.

**Note:** This parameter is valid only if DTAOPT(\*LIB) is specified.

**\*NO:** The PRTTTC (Print Trace) command is not run as part of this command.

**\*YES:** The PRTTTC (Print Trace) command is run after the trace data has been stored in the database files.

## Examples for ENDTRC

### Example 1: End Most Recently Started Trace

```
ENDTRC SSNID(*PRV)
```

This command ends the trace session started most recently by the same user who is running the ENDTRC command. The trace data will be stored in a set of files in the current library of the job, or QGPL if there is no current library for the job.

### Example 2: End a Trace and Delete Trace Data

```
ENDTRC SSNID(DCG1) DTAOPT(*DLT)
```

This command ends the trace session DCG1 and deletes the trace data.

## Error messages for ENDTRC



### **\*ESCAPE Messages**

#### **CPC3923**

ENDTRC session ID &1 successfully saved into library &2.

#### **CPC3924**

ENDTRC session ID &1 successfully deleted.

#### **CPF39CA**

Trace session ID &1 not found.

#### **CPF39CB**

Trace session ID &1, in library &2, data exists. Specify RPLDTA(\*YES).

---

## **ENDTRPMGR (End Trap Manager) Command Description**

ENDTRPMGR Command syntax diagram

### **Purpose**

The End Trap Manager (ENDTRPMGR) command allows you to end the OS/400 SNMP Manager Framework trap manager job.

There are no parameters for this command.

### **Example for ENDTRPMGR**

```
ENDTRPMGR
```

This command ends the OS/400 SNMP Manager Framework trap manager job.

### **Error messages for ENDTRPMGR**

### **\*ESCAPE Messages**

#### **CPFA805**

Trap manager job not active or being ended.

---

## **ENDUSF (End Ultimedia System Facilities) Command Description**

ENDUSF Command syntax diagram

### **Purpose**

The End Ultimedia System Facilities (ENDUSF) command ends the Ultimedia Facilities server and router programs, QUMBVSERVER and QUMBVROUTR, in the QSYSWRK subsystem of the operating system.

There are no parameters for this command.

### **Example for ENDUSF**

```
ENDUSF
```

This command ends the Ultimedia System Facilities program.

No error messages.

---

## ENDWTR (End Writer) Command Description

ENDWTR Command syntax diagram

### Purpose

The End Writer (ENDWTR) command ends the specified spooling writer and makes its associated output device available to the system. The writer can be ended immediately or in a controlled manner. If it is ended immediately, the writer stops writing the file and the file is again available on the output queue. If it is ended in a controlled manner, the writer finishes writing the current file (or a copy of a file), or it finishes printing a page of the file, if it is a printer writer, before it is ended.

### Required Parameter

**WTR** Specifies the name of the spooling writer being ended. The writer's output device is available to the system.

**\*ALL:** All writers that are started are ended.

**\*SYSVAL:** The writer started to the system default printer is ended.

*writer-name:* Specify the name of the writer being ended.

### Optional Parameter

#### OPTION

Specifies when the writer stops processing. Output stops at the end of the spooled file (or copy of a file) currently being written to an output device.

**\*CNTRLD:** The job is ended in a controlled manner. This allows the program to perform cleanup (end-of-job processing).

#### Note:

The spooled file that is currently being processed remains on the output queue. If the printer is a printer writer, printing stops and a form feed is done.

**\*IMMED:** The operation ends immediately.

**\*PAGEEND:** The writer is stopped after processing of the current buffer. This value is valid only if the spooling writer is a printer writer.

### Example for ENDWTR

```
ENDWTR WTR(PRINTER)
```

This command stops the writer named PRINTER at the end of the spooled file whose output is being printed, and then releases the device to the system.

### Error messages for ENDWTR

#### \*ESCAPE Messages

##### CPF1317

No response from subsystem for job &3/&2/&1.

##### CPF1340

Job control function not performed.

##### CPF1352

Function not done. &3/&2/&1 in transition condition.

**CPF1842**

Cannot access system value &1.

**CPF3313**

Writer &1 not active nor on job queue.

**CPF3330**

Necessary resource not available.

**CPF3331**

Not authorized to control writer &3/&2/&1.

**CPF3339**

Previous end request to writer &3/&2/&1 pending.

**CPF3438**

\*PAGEEND not valid for writer &3/&2/&1.

---

## EXTMEDIBRM (Extract Media Information) 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.

EXTMEDIBRM Command syntax diagram

### Purpose

The Extract Media Information (EXTMEDIBRM) command adds information about a volume to the BRMS media inventory contents. You specify the identifier of the volume and what is contained on the volume. BRMS records content information only for media already added to the BRMS media inventory, and then only for media whose contents are currently shown as expired. If active contents information for the media already exists, or if the media has not been added to the media inventory, no content information is added.

### Example for EXTMEDIBRM

#### Example 1: Adding Media Information

```
EXTMEDIBRM DEV(TAP06) FILE(*SAVLIB)
```

In this example, the volume that is mounted on device TAP06 has save library content information added to the BRMS media inventory.

### Error messages for EXTMEDIBRM

None >>

---

## GENLICENSE (Generate License Key) Command Description

**Note:** To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

GENLICENSE Command syntax diagram

### Purpose

The Generate License Key (GENLICENSE) command generates a license key to enable users to access a product or a feature of a product. This key is specific to the product and system information entered in this

command. The resulting key is a combination of 18 characters and numbers ranging from A-F and 0-9. To run this command, the product definition of the product you are generating the key for must exist on the system.

This command also adds the license information to the license repository. The keys are saved in the repository to keep a history of all the keys generated. You can display the repository to see what keys were generated for a specific product or system. To work with the license repository, you can use the DSPLICKEY, ADDLICKEY, and RMVLICKEY commands.

You can use the Add License Key Information (QLZAADDK) API or the Add Product License Information (QLZADDLI) API to add license information. For more specific details about these APIs, see the Application Program Interfaces (APIs) topic in the Information Center.

**Restriction:** This command is shipped with public \*EXCLUDE authority.

### Required Parameters

#### PRDID

Specifies the 7-character identifier (ID) of the product for which the license key is to be generated.

#### LICTRM

Specifies the license term for which the license key is to be generated.

*license-term:* Specify the license term of the product in one of the following formats:

Vx

VxRy

VxRyMz

where values for x and y range from 0 through 9, and values for z range from 0 through 9 and from A through Z. A license term of Vx means the authorized usage limit is valid for the entire version of the product or feature. A license term of VxRy means the authorized usage limit is valid for the entire release of the product or feature. A license term of VxRyMz means the authorized usage limit is valid only for a modification of the product.

#### FEATURE

Specifies the feature of the product specified on the PRDID parameter for which the license key is to be generated.

*feature:* Specify the feature for which a license key is to be generated. Valid values range from 5001 through 9999.

#### VNDPWD

Specifies the software vendor's password. This password is encrypted and stored with the product. It is used to validate the generate license key request and must be the same password you use on the Add Product License Information (ADDPRDLICI) command.

**\*NONE:** There is no vendor password needed to generate a license key.

*vendor-password:* Specify the vendor password. It must begin with an alphabetic character (A through Z, \$, #, or @) followed by no more than 9 alphanumeric characters (A through Z, 0 through 9, \$, #, @, or \_).

#### SERIAL

Specify the system serial number of the system the license key is being generated for.

*system-serial-number:* Specifies the system serial number of the system the license key is being generated for.

#### PRCGRP

Specifies the processor group of the system the license key is being generated for.

**\*ANY:** The license key being generated works with any processor group.

*processor-group:* Specify the processor group of the system the license key is being generated for. Use the Work with License Information (WRKLICINF) command on the system the key is being generated for to display the processor group.

## Optional Parameters

### EXPDATE

Specifies the date the product license expires. After this date, users over the default usage limit are not allowed to access the product or feature. A new license key must be obtained from the software vendor to allow further use of the product.

**\*NONE:** There is no expiration date for the product or feature.

*expiration-date:* Specify the date that the product or feature expires.

### USGLMT

Specifies the maximum number of users (usage limit) for the product or feature for which the license key is being generated. For concurrent usage, specifies the maximum number of jobs allowed to access the product or feature at the same time. For registered usage, this is the maximum number of license users that can be registered to use this product or feature.

**1:** One user is allowed to access the product or feature.

**\*NOMAX:** The number of users is not limited.

*usage-limit:* Specify the maximum number of users for this product or feature. Valid values range from 0 through 999999.

### VNDDTA

Specifies an 8-character data field to be defined and used by the vendor.

**\*NONE:** There is no vendor data for this product or feature.

*vendor-data:* Specifies 8 characters that help define further information on the product or feature.

### OUTPUT

Specifies whether the license key that is generated is displayed in a completion message, printed in a spooled file, or saved in a file.

**\***: The license key is only shown in a completion message.

**\*PRINT:** The license key is printed in a spooled file.

**\*LICKEYFILE:** The license key information is saved to the file specified on the LICKEYFILE parameter. This file can then be used as a key file for Distributed Systems License Option (DSLO) tapes.

### LICKEYFILE

Specifies the file in which the license key information is saved. If this file does not exist, the system creates it. If it does exist, it must be in the format of QSYS/QALZKEY.

The possible library values are:

**\*LIBL:** The library list is used to locate the file in which to save the license key.

**\*CURLIB:** The current job library is used to locate the file in which to save the license key. If no library is specified as the current library, the QGPL library is used.

*library-name:* Specify the name of the library where the license key is to be saved.

*file-name*: Specify the name of the file into which the license key information is to be saved.

## LICKEYMBR

Specifies which member the license key information is saved into and how the information is saved.

### Note:

You can also use the Generate License Key (QLZAGENK) API to create the license key information. See the Application Program Interfaces (APIs) topic in the Information Center for more information about this API.

### Element 1: Member to Receive Output

**\*FIRST:** The first member in the file receives the output. If LICKEYMBR(\*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the LICKEYFILE parameter.

*member-name*: Specify the name of the file member that receives the output. If the name of the file member is specified and the member does not exist, the system creates it. If the member already exists, you have the option of either adding a new record to the end of the existing member or clearing the member and then adding the new record.

### Element 2: Option to Perform on Member

**\*ADD:** If the member exists, the system adds the new record to the end of the existing records.

**\*REPLACE:** If the member exists, the system clears it and then adds the new record.

### Example for GENLICKEY

```
GENLICKEY PRDID(2MYPROD) LICTRM(V5R2M0) FEATURE(5001)
  VNDPWD(PRODUCTPWD) EXPDATE(*NONE) SERIAL(1234567)
  PRCGRP(P30) USGLMT(25)
```

This command generates a license key for product 2MYPROD, license term V5R2M0 and feature 5001. The key allows 25 users to be authorized to use this product and there is no expiration date. The key allows the product to only be used on a system with a serial number of 1234567 and a processor group of 01 and a processor group of P30 or less.

### Error messages for GENLICKEY

#### \*ESCAPE Messages

None 

---

## GOTO (Go To) Command Description

GOTO Command syntax diagram

### Purpose

The Go To (GOTO) command is used in CL programs for branching from one part of the program to another. The branching is to the label on another command that is specified on the GOTO command. Branching can be either forward or backward, but the specified label must be inside the program.

**Restriction:** This command is valid only within a CL program.

## Required Parameter

### CMDLBL

Specifies the label name of the command to which control is transferred when the GOTO command is processed. The command with the label is then processed. If the specified command cannot be run (for example, if it is a DCL command), control is transferred to the next command following the command with the specified label. The label must be within the same program as the GOTO command. A CL variable name cannot be used to specify the label name.

### Example for GOTO

```
LOOP: CHGVAR &A (&A + 1)
      IF (&A *LT 30) THEN(GOTO LOOP)
```

The Change Variable (CHGVAR) command increases the value of &A by 1 until &A is equal to or greater than 30. The GOTO command is processed each time that the IF command tests the expression and the result is true; the GOTO command following the THEN parameter causes the program to branch back to the label LOOP on the CHGVAR command. Refer to the descriptions of the CHGVAR command and the IF command for additional explanations of their functions.

### Error messages for GOTO

None

---

## GO (Go to Menu) Command Description

GO Command syntax diagram

### Purpose

The Go to Menu (GO) command allows the user to specify either a particular menu or a generic menu name. The user may optionally specify whether or not to return to the menu or display from which the command is entered after running the menu specified by the GO command.

### Using the Previous and Exit Keys

A menu is placed on an internal menu stack before it is run. If a stack is not available for the menu, one is created. When the Cancel key is pressed for a menu, the previous menu in the stack is shown. Each menu stack is ten elements (menus) deep. When the eleventh menu is placed on the menu stack, the first, or oldest, menu is removed from the stack. This menu cannot be returned to by using the Cancel key.

Pressing the Exit key returns the user to the last display or menu from which a GO command was entered with RTNPNT(\*YES). (Note that \*YES is the default value for the optional RTNPNT parameter.) The display that the user is returned to is found by removing menus from the menu stack until a return point is found. This process may also cause a program in the call stack to return to its calling program unless the program is a return point.

Pressing either the Exit or Cancel key while viewing help for a menu returns the user to the menu.

### Restrictions:

1. The user must have \*USE authority for the menu and its display and message files or program (whichever applies) to use this command.
2. The user must also have \*USE authority for the library where the menu is located.

### Required Parameter

**MENU** Specifies the qualified name of the menu being run.

The name of the menu 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.

**\*USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

**\*ALL:** All libraries in the system, including QSYS, are searched.

» **\*ALLUSR:** User libraries are all libraries with names that do not begin with the letter Q except for the following:«

#CGULIB	#DSULIB	#SEULIB
#COBLIB	#RPGLIB	
#DFULIB	#SDALIB	

» Although the following libraries with names that begin with the letter Q are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are also considered user libraries:«

QDSNX	» QSYS2xxxx«	QUSROND
QGPL	QS36F	QUSRPOSGS
QGPL38	QUSER38	QUSRPOSSA
QMPGDATA	QUSRADSM	QUSRPYMSVR
QMQMATA	QUSRBRM	QUSRRDARS
QMQMPROC	QUSRDIRCL	QUSRSYS
QPFRDATA	QUSRDIRDB	QUSRVI
QRCL	QUSRIJS	QUSRVxRxMx
» QRCLxxxx«	QUSRINFSKR	
» QSYS2«	QUSRNOTES	

#### Notes:

- » 'xxxx' is the number of a primary auxiliary storage pool.«
- A different library name, of the form QUSRVxRxMx, can be created by the user for each release that IBM supports. VxRxMx is the version, release, and modification level of the library.

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

**\*ALL:** A list of all menus in the given library are presented to allow the selection of the menu to be run.

*menu-name:* Specify the name of the menu being run.



*generic\*-menu-name*: Specify the generic name of the menu. 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. If the complete object name is specified, and multiple libraries are searched, multiple objects can be specified only if \*ALL or \*ALLUSR library values can be specified for the name. For more information on the use of generic names, refer to generic names.

## Optional Parameter

### RTNPNT

Specifies whether the display where the command is entered is returned to when the Exit key is pressed.

**\*YES:** The display where the command is entered is shown when the Exit key is pressed.

**\*NO:** The display where the command is entered is not shown when the Exit key is pressed.

### Example for GO

```
GO MENU(PERSMENU)
```

This command runs a menu called PERSMENU, located in a library found by searching the library list (\*LIBL default value).

If the Exit key is pressed while PERSMENU is being shown, the display where the GO command was entered (\*YES default value on the RTNPNT parameter) is shown.

### Error messages for GO

#### \*ESCAPE Messages

##### CPF6ACD

Menu &1 in &2 is wrong version for system.

##### CPF6AC7

Menu &1 in library &2 not displayed.

---

## GRTACCAUT (Grant Access Code Authority) Command Description

GRTACCAUT Command syntax diagram

### Purpose

The Grant Access Code Authority (GRTACCAUT) command gives the specified users authority to access documents and folders associated with the access codes. Access is restricted to read only (\*USE authority).

### Restrictions:

1. This command is shipped with public \*EXCLUDE authority.
2. The access code must be defined to the system (by using the Add Access Code (ADDACC) command) before the user can give access code authority.
3. The user being given access code authority must be enrolled in the system distribution directory.

**Note:**

When a new user profile is created based on an existing user profile, the access codes (if any) associated with the user do not carry over to the new user. The GRTACCAUT command must be used for the new user profile; however, the REFUSER parameter can be specified on this command to get all the codes of the user who is being based on.

Authority for this command is originally restricted to the security officer. Authority to use this command can be granted to other users with the Grant Object Authority (GRTOBJAUT) command.

**Required Parameters**

**ACC** Specifies the access code that is being authorized for use by the user identified on the USER parameter.

**\*REFUSER:** The access code authority granted is based on a second user profile name; that name must be specified on the REFUSER parameter.

*access-code:* Specify a number, ranging from 1 through 2047, that specifies the access code of the documents and folders for which the user wants authority to be granted. The access code must be defined to the system using the ADDACC command before being specified in this parameter. Up to 300 access codes can be specified.

**USER** Specifies the user profile name of the users to whom access code authority is granted. The user identified has the access code added to the list of access codes to which the user is authorized; this access code is used to verify additional document and folder accesses from the document library. The user must be enrolled in the system distribution directory before being granted authority to use an access code. Specify the name of the user profile. Up to 300 can be specified.

**Optional Parameter****REFUSER**

Specifies the user referred to, which is the user profile on which to base the access code authority. If this parameter is used, \*REFUSER must be specified on the ACC parameter. The \*REFUSER's access code authority is added to the existing access code authority specified on the USER parameter. All access code authority of the \*REFUSER is copied except for the access code authority the \*REFUSER's group profile may have.

**\*NONE:** No user is referred to.

*based-on-user-profile:* Specify the user-profile name on which the access code authority is based. This user must also be enrolled in the system distribution directory.

**Examples for GRTACCAUT****Example 1: Granting Authority to Multiple Users**

```
GRTACCAUT ACC(3 30 60) USER(SAM LARRY)
```

This command gives authority to access codes 3, 30, and 60 to SAM and LARRY.

**Example 2: Granting Authority Based on Another User**

```
GRTACCAUT ACC(*REFUSER) USER(JOE) REFUSER(TOM)
```

This command grants access code authority to JOE based on TOM's authority. For example, if JOE currently has authority to access codes 1, 12, and 50, and TOM currently has authority to access codes 8 and 9, the GRTACCAUT command authorizes JOE to access codes 1, 8, 9, 12, and 50.

## Error messages for GRTACCAUT

### \*ESCAPE Messages

#### CPF9009

System requires file &1 in &2 be journaled.

#### CPF9013

Access code authority given to &1 users, not granted to &2 users.

#### CPF9024

System cannot get correct record to finish operation.

#### CPF9065

Not allowed to give access code authority.

#### 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.

---

## GRTOBJAUT (Grant Object Authority) Command Description

GRTOBJAUT Command syntax diagram

### Purpose

The Grant Object Authority (GRTOBJAUT) command is used by one user to grant specific authority for the object named in this command to another user or group of users.

Authority can be given to:

- Named users
- Users (\*PUBLIC) who do not have authority specifically given to them either for the object or for the authorization list
- Users of the referenced object (specified in the REFOBJ parameter)
- Users on an established authorization list

If AUT(\*AUTL) is specified, the PUBLIC authority for the object comes from the PUBLIC authority of the authorization list securing the object.

The AUTL parameter is used to secure an object with an authorization list. User profiles cannot be secured by an authorization list (\*AUTL).

This command can be used by an object's owner or by a user with object management authority for the specified object. A user with object management authority can grant to other users any authority that user has, except object management authority. Only the owner of the object, or someone with all object special authority (\*ALLOBJ), can grant object management authority to a user.

A user with \*ALL authority can assign a new authorization list.

When granting authority to users, the REPLACE parameter indicates whether the authorities you specify replace the user's existing authorities. The default value of REPLACE(\*NO) gives the authority that you

specify, but it does not remove any authority that is greater than you specified, unless you are granting \*EXCLUDE authority. REPLACE(\*YES) removes the user's current authorities, then grants the authority that you specify.

When granting authority with a reference object, this command gives the authority that you specify, but it does not remove any authority that is greater than you specified, unless you are granting \*EXCLUDE authority.

### Restrictions:

1. This command must get an exclusive lock on a database file before read or object operational authority can be given to a user.
2. If a user requests authority for another specified user to a device currently in use by another authorized user, authority to the device is not given.
3. Object type \*AUTL cannot be specified.
4. AUT(\*AUTL) is valid only with USER(\*PUBLIC).
5. A user must either be the owner of the object or have \*ALL authority to use the AUTL parameter.
6. The user must have object management authority to the object.
7. If the object is a file, the user must have object operational and object management authorities.
8. For display stations or for work station message queues associated with the display station, if this command is not entered at the device for which authorities are being granted, it should be preceded by the Allocate Object (ALCOBJ) command and followed by the Deallocate Object (DLCOBJ) command.
9. >> You must have \*USE authority to the auxiliary storage pool device if one is specified. <<

### Required Parameters

**OBJ** Specifies the qualified name of the objects for which specific authorities are given to one or more users or to an authorization list. A specific object name or a generic object name can be qualified by a library name. More information on this parameter is in commonly used parameters.

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.

**\*USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

**\*ALL:** >> All libraries in the auxiliary storage pools (ASPs) specified by the ASPDEV parameter are searched. <<

**\*ALLUSR:** >> All user libraries in the ASPs specified by the ASPDEV parameter are searched. << All libraries with names that do not begin with the letter Q are searched except for the following:

#CGULIB	#RPGLIB
#COBLIB	#SDALIB
#DFULIB	#SEULIB
#DSULIB	

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are also considered user libraries and are also searched:

QDSNX	QS36F	QUSROND
QGPL	QUSER38	QUSRPOSGS
QGPL38	QUSRADSM	QUSRPOSSA
QMPGDATA	QUSRBRM	QUSRPYMSVR
QMQMDATA	QUSRDIRCL	QUSRRDARS
QMQMPROC	QUSRDIRDB	QUSRSYS
QPFRDATA	QUSRIJS	QUSRVI
QRCL	QUSRINFSKR	QUSRVRxRxMx
» QRCLnnnnn «	QUSRNOTES	

#### Notes:

1. » “nnnnn” is the number of a primary auxiliary storage pool. «
2. A different library name, of the form QUSRVRxRxMx, can be created by the user for each release that IBM supports. VxRxMx is the version, release, and modification level of the library.

» **\*ALLAVL:** All libraries in all available ASPs are searched.

**\*ALLUSRAVL:** All user libraries in all available ASPs are searched. Refer to **\*ALLUSR** for a definition of user libraries. «

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

**\*ALL:** All objects of the specified type (OBJTYPE) found in the search have specific authorities granted. A specific library name must be specified.

*object-name:* Specify the name of the object for which specific authorities are given to one or more users.

*generic\*-object-name:* Specify the generic name of the object. 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. If the complete object name is specified, and multiple libraries are searched, multiple objects can be granted only if **\*ALL**, **\*ALLUSR**, » **\*ALLAVL**, or **\*ALLUSRAVL** « library values can be specified for the name. For more information on the use of generic names, refer to generic names.

#### OBJTYPE

Specifies the object type of the object for which specific authorities are given to the specified users or to an authorization list. More information on this parameter is in commonly used parameters.

**\*ALL:** Specific authorities for all object types (except **\*AUTL**) are given to the specified users or to the authorization list.

*object-type:* Specify the specific object type of the object for which specific authorities are given to the specified users.

**USER** Specifies the user names of one or more users to whom authorities for the named object are being given. If user names are specified, the authorities are given specifically to those users. Authority given by this command can be revoked specifically by the Revoke Object Authority (RVKOBJAUT) command.

**\*PUBLIC:** Users are authorized to use the object as specified in the AUT parameter when they do not have authority specifically given to them for the object, are not on the authorization list and none of their groups have any authority or are not on the authorization list. Users who do not have any authority, and whose groups do not have any authority, are authorized to use the object as specified in the AUT parameter.

*user-profile-name:* Specify the user names of one or more users to have specific authority for the object. Up to 50 user profile names can be specified.

**AUTL** Specifies the name of the authorization list whose users are given authority for the object specified in the OBJ parameter.

**\*NONE:** The object specified on the OBJ parameter will no longer be secured by an authorization list. If public authority to the object is \*AUTL, it is changed to \*EXCLUDE.

*authorization-list-name:* Specify the name of the authorization list whose users are given authority for the object specified on the OBJ parameter.

## REFOBJ

Specifies the name of the object being queried to obtain authorization information. Those authorizations are given to the object specified by the OBJ parameter. Users authorized to the referenced object are authorized in the same manner to the object for which authority is being given. If the referenced object is secured by an authorization list, that authorization list secures the object specified in the OBJ parameter. Specify the name of the object.

The name of the reference 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.

## Optional Parameters

**AUT** Specifies the authority given to users specified on the USER parameter. Users must have \*AUTLMGT authority to manage the authorization list. More information on this parameter is in commonly used parameters.

**\*CHANGE:** The user can perform all operations on the object except those limited to the owner or controlled by object existence authority and object management authority. The user can change and perform basic functions on the object. Change authority provides object operational authority and all data authority.

**\*ALL:** The user can perform all operations except those limited to the owner or controlled by authorization list management authority. The user can control the object's existence, specify the security for the object, change the object, and perform basic functions on the object. The user also can change ownership of the object.

**\*USE:** The user can perform basic operations on the object, such as run a program or display the contents of a file. The user is prevented from changing the object. Use authority provides object operational authority, read authority, and execute authority.

**\*EXCLUDE:** The user cannot access the object.

**\*AUTL:** The public authority of the authorization list specified in the AUTL parameter is used for the public authority for the object.

A maximum of ten of the following values can be specified:

**\*OBJALTER:** Object alter authority provides the authority needed to alter the attributes of an object. If the user has this authority on a database file, the user can add and remove triggers, add and remove referential and unique constraints, and change the attributes of the database file. If the user has this authority on an SQL package, the user can change the attributes of the SQL package. This authority is currently only used for database files and SQL packages.

**\*OBJEXIST:** Object existence authority provides the authority to control the object's existence and ownership. This authority is necessary for users who want to delete the object, free storage of the object, perform save and restore operations for the object or transfer ownership of an object. (If a user has special save system authority (\*SAVSYS), object existence authority is not required.) Object existence authority is required to create an object that has been named by an authority holder.

**\*OBJMGT:** Object management authority provides the authority to specify the security for the object, move or rename the object, and add members to database files.

**\*OBJOPR:** Object operational authority provides authority to look at the description of an object and use the object as determined by the data authorities that the user has to the object.

**\*OBJREF:** Object reference authority provides the authority needed to reference an object from another object such that operations on that object may be restricted by the other object. If the user has this authority on a physical file, the user can add referential constraints in which the physical file is the parent. This authority is currently only used for database files.

**\*ADD:** Add authority provides the authority to add entries to an object (for example, job entries to a queue or records to a file).

**\*DLT:** Delete authority allows the user to remove entries from an object, for example, remove messages from a message queue or records from a file.

**\*EXECUTE:** Execute authority provides the authority needed to run a program or locate an object in a library.

**\*READ:** Read authority provides the authority needed to get the contents of an entry in an object.

**\*UPD:** Update authority provides the authority needed to change the entries in an object.

## REFOBJTYPE

Specifies the object type of the referenced object (REFOBJ parameter).

**\*OBJTYPE:** The object type of the referenced object is the same type as the object being given authority (OBJTYPE parameter).

*object-type:* Specify the type of the object. Any one of the operating system object types can be specified.

## REPLACE

Specifies whether the authorities replace the user's current authorities.

**\*NO:** The authorities are given to the user, but no authorities are removed, unless you are granting \*EXCLUDE authority.

**\*YES:** The user's current authorities are removed, then the authorities are given to the user.

## » 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 grant 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.

## REFASPDEV

Specifies the auxiliary storage pool (ASP) device name where the library that contains the reference object (REFOBJ parameter) is located. If the reference 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 queried for authorities.

**\***: The ASPs that are currently part of the job's library name space will be searched to locate the reference 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 user basic ASPs will be searched to locate the reference 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 reference 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. «

## Examples for GRTOBJAUT

### Example 1: Granting Authority to All Users

```
GRTOBJAUT OBJ(USERLIB/PROGRAM1) OBJTYPE(*PGM)
USER(*PUBLIC)
```

This command gives authority to use the object named PROGRAM1 to all users of the system who do not have authorities specifically given to them, who are not on an authorization list, whose user groups do not have authority to the object, or whose user groups are not on the authorization list. The object is a program (\*PGM) located in the library named USERLIB. Because the AUT parameter is not specified, the authority given to all users is change authority. This allows all users to run the program and to debug it.

### Example 2: Granting Object Management Authority

```
GRTOBJAUT OBJ(ARLIB/PROGRAM2) OBJTYPE(*PGM)
USER(TMSMITH) AUT(*OBJMGT)
```

This command gives object management authority to user named TMSMITH. This authority allows TMSMITH to grant to others personally possessed authorities for the object named PROGRAM2, which is a program located in the library named ARLIB.

### Example 3: Granting Authority to Users on Authorization List



```
GRTOBJAUT OBJ(MYLIB/PRGM3) OBJTYPE(*PGM)
AUTL(KLIST)
```

This command gives to users the authority specified for them on authorization list KLIST for the object named PRGM3. The object is a program located in library MYLIB.

## **Error messages for GRTOBJAUT**

### **\*ESCAPE Messages**

#### **CPF22A0**

Authority of \*AUTL is allowed only with USER(\*PUBLIC).

#### **CPF22A1**

OBJTYPE(\*AUTL) not valid on this command.

#### **CPF22A2**

Authority of \*AUTL not allowed for object type \*USRPRF.

#### **CPF22A3**

AUTL parameter not allowed for object type \*USRPRF.

#### **CPF22A9**

Authority of \*AUTL cannot be specified.

#### **CPF22DA**

Operation on file &1 in &2 not allowed.

#### **» CPF22F0**

Unexpected errors occurred during processing. «

#### **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.

#### **CPF2210**

Operation not allowed for object type \*&1.

#### **CPF2211**

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

#### **CPF2216**

Not authorized to use library &1.

#### **CPF2223**

Not authorized to give authority to object &1 in &3 type \*&2.

#### **CPF2227**

One or more errors occurred during processing of command.

#### **CPF2236**

AUT input value not supported.

#### **CPF2243**

Library name &1 not allowed with OBJ(generic name) or OBJ(\*ALL).

#### **CPF2245**

Process profile not owner of object &1 in &3 type \*&2.

**CPF2253**

No objects found for &1 in library &2.

**CPF2254**

No libraries found for &1 request.

**CPF2273**

Authority may not have been changed for object &1 in &3 type \*&2 for user &4.

**CPF2283**

Authorization list &1 does not exist.

**CPF2290**

\*EXCLUDE cannot be specified with another authority.

**» CPF980B**

Object &1 in library &2 not available. «

**CPF9804**

Object &2 in library &3 damaged.

**» CPF9814**

Device &1 not found.

**CPF9825**

Not authorized to device &1.

**CPF9873**

ASP status is preventing access to object. «

---

## GRTUSRAUT (Grant User Authority) Command Description

GRTUSRAUT Command syntax diagram

### Purpose

The Grant User Authority (GRTUSRAUT) command grants authority to a user by referring to another user profile. Authorization lists or group profiles should be used instead of GRTUSRAUT command support if possible. The GRTUSRAUT command can take a long time to run, and the time to run a subsequent SAVSYS or SAVSECDTA command also increases. The authority granted to the user depends on the user profile being referred to and the user using the command.

If the security officer enters this command, the user specified in the USER parameter is granted the same authority for each object as granted in the user profile being referred to, including object management authority.

If the user whose user profile is specified for the REFUSER parameter enters this command, the user specified for the USER parameter is given all authorities for each object owned by the referenced profile, including object management authority.

For objects that the user profile being referred to does not own but is authorized to use, the user of this command must have object management authority and the authorities being granted for the object, or must own the object. Otherwise, no authority is granted for that object. Object management authority is not granted unless the user of this command owns the object being authorized.

Ownership of an object or authorities held by the referred to user cannot be changed by this command. Authorities to objects granted to a user profile are added to any authorities that the user profile already has.

### Restriction:

The following user profiles cannot be specified for either of the parameters on this command: >>

QAUTPROF	QGATE	QSPL
QCLUMGT	QIPP	QSYS
QCLUSTER	QLPAUTO	QTCM
QCOLSRV	QLPINSTALL	QTCP
QDBSHR	QMSF	QTMHHTP1
QDBSHRDO	QNETSPLF	QTMHHTP
QDFTOWN	QNFSANON	QTSTRQS
QDIRSRV	QNTP	QYPSJSVR
QDLFM	QPEX	
QDOC	QPM400	
QDSNX	QRJE	
QEJB	QSNADS	



### Required Parameters

**USER** Specifies the name of the user profile to which authority is being granted.

### REFUSER

Specifies the name of the user profile being referred to for authority.

### Examples for GRTUSRAUT

#### Example 1

User SECOFR is entering this command:

```
GRTUSRAUT USER(USRB) REFUSER(USRA)
```

This command grants the user profile USRB the same authorities that USRA has for all objects that USRA owns (including object management authority) or has authority to.

#### Example 2

User USRA is entering this command:

```
GRTUSRAUT USER(USRB) REFUSER(USRC)
```

This command grants the user profile USRB the same authorities that USRC has for all objects that USRC has authorities to only if USRA, entering this command, has object management authority to the objects or is the owner of the objects being referred to.

### Error messages for GRTUSRAUT

#### \*ESCAPE Messages

##### CPF2204

User profile &1 not found.

##### CPF2211

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

##### CPF2213

Not able to allocate user profile &1.

##### CPF2217

Not authorized to user profile &1.

##### CPF2222

Storage limit is greater than specified for user profile &1.

**CPF2223**

Not authorized to give authority to object &1 in &3 type \*&2.

**CPF2252**

Authority given to &2 objects. Authority not given to &3 objects.

---

## GRTUSRPMN (Grant User Permission) Command Description

GRTUSRPMN Command syntax diagram

### Purpose

The Grant User Permission (GRTUSRPMN) command gives permission to a user to handle documents and folders or to perform tasks on behalf of another user. Access is restricted to documents, folders, and mail that is not personal. For example, user A gives user B the ability to work on behalf of user A. User B is now allowed to obtain, file, delete, retrieve, list, distribute, cancel, and search a document library on behalf of user A; however, user B is not allowed to obtain any of user A's personal mail or get any of user A's private documents or folders from the library.

The users specified must be enrolled in the system distribution directory before using this command. To enroll a user, use the Work with Directory Entries (WRKDIRE) command.

**Restriction:** The user must have \*ALLOBJ authority to grant permission for a user to work on behalf of another user.

Additional information on giving document authority is in the SNA Distribution Services  book.

### Required Parameter

**TOUSER**

Specifies the name of the user profile that is permitted to work on behalf of the user specified in the FORUSER parameter. Access is restricted to objects that are not personal. The user profile must exist at the time the command is run, and the user must be enrolled in the system distribution directory before running the command.

### Optional Parameter

**FORUSER**

Specifies the name of the user profile that the user specified in the TOUSER parameter works on behalf of. The user must be enrolled in the system distribution directory before running the command.

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

*user-profile-name:* Specify the name of the user profile.

### Example for GRTUSRPMN

```
GRTUSRPMN TOUSER(JUDY) FORUSER(PEGGY)
```

JUDY is the administrative assistant for an executive. This command allows JUDY to work with documents or folders for PEGGY that are not personal.

### Error messages for GRTUSRPMN

#### \*ESCAPE Messages

**CPF9007**

User permission given for &1 users, not given &2 users.

**CPF9009**

System requires file &1 in &2 be journaled.

**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.

---

## **GRTWSOAUT (Grant Workstation Object Authority) Command Description**

GRTWSOAUT Command syntax diagram

**Purpose**

The Grant Workstation Object Authority (GRTWSOAUT) command is used by one user to grant specific authority for the workstation object named in this command to another user or group of users. Workstation objects are used by the OS/400 Graphical Operations program.

Authority can be given to:

- Named users
- Users (\*PUBLIC) who do not have authority specifically given to them either for the object or for the authorization list
- Groups of users who do not have any authority to the object or are not on the authorization list that secures the object
- Users of the referenced workstation object (specified on the REFWSO parameter)
- Users on an established authorization list

When AUT(\*AUTL) is specified, the user can specify the authority for:

- All users who do not have authority specifically given to them for an object.
- Users who are not on the authorization list that secures the object.
- Users whose user group does not have authority specifically given to it.
- Users whose user group is not on the authorization list that secures the object.

This command can be used by an object owner, by the security officer, or by a user with object management authority for the specified object.

**Restrictions:**

1. A user must be either the owner of the object or have \*ALL authority to use the AUTL parameter.
2. The user must have object management authority to the object to grant authority to the object.
3. AUT(\*AUTL) can be specified only with USER(\*PUBLIC). User profile names cannot be secured by an authorization list (\*AUTL).
4. Only the owner of the object, or someone with all object authority (\*ALLOBJ), can grant object management authority to a user.

**Required Parameters****WSOTYPE**

Specifies the name of the workstation object for which specific authorities are given to one or more users or to an authorization list.

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
*TPLMSGSEND	Message sender template
*MSGSEND	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
*PRSET	Personal setting objects

**USER** Specifies the user profile names of one or more users to whom authorities for the named object are being given. If user names are specified, the authorities are given specifically to those users. Authority given by this command can be revoked specifically by the Revoke Workstation Object Authority (RVKWSOAUTH) command.

**\*PUBLIC:** All users of the system, who do not have authority specifically given to them for the object, who are not on the authorization list, whose user group does not have any authority, or whose user group is not on the authorization list, are authorized to use the object as specified on the AUT parameter.

*user-profile-name:* Specify the user profile names of one or more users who have specific authority for the object. A maximum of 50 user profile names can be specified.

**AUTL** Specifies the name of the authorization list whose members are given authority for the object specified on the WSOTYPE parameter.

## REFWSO

Specifies the name of the workstation object being queried to obtain authorization information. Those authorizations are given to the object specified on the WSOTYPE parameter. Users authorized to the referenced object are authorized in the same manner to the object for which authority is being given. If the referenced object is secured by an authorization list, that authorization list secures the object specified on the WSOTYPE parameter. Specify the name of the object.

## Optional Parameter

**AUT** Specifies the authority given to users specified on the USER parameter. Users must have \*AUTLMGT authority to manage the authorization list.

**\*CHANGE:** The user can perform all operations on the object except those limited to the owner or controlled by object existence authority and object management authority. The user can change and perform basic functions on the object. Change authority provides object operational authority and all data authority.

**\*ALL:** The user can perform all operations except those limited to the owner or controlled by authorization list management authority. The user can control the object's existence, specify the security for the object, change the object, and perform basic functions on the object. The user also can change ownership of the workstation object.

**\*USE:** The user can perform basic operations on the workstation object, such as running a program or reading a file. The user cannot change the workstation object. \*USE authority provides object operational authority, read authority, and execute authority.

**\*EXCLUDE:** The user cannot access the workstation object.

**\*AUTL:** The public authority of the authorization list specified on the AUTL parameter is used for the public authority for the object.

**A maximum of ten of the following values can be specified:**

**\*OBJALTER:** Object alter authority provides the authority needed to alter the attributes of an object. If the user has this authority on a database file, the user can add and remove triggers, add and remove referential and unique constraints, and change the attributes of the database file. If the user has this authority on an SQL package, the user can change the attributes of the SQL package. This authority is currently only used for database files and SQL packages.

**\*OBJEXIST:** Object existence authority provides the authority to control the object's existence and ownership. This authority is necessary for users who want to delete the object, free storage of the object, perform save and restore operations for the object, or transfer ownership of the object. (If a user has special save system authority (\*SAVSYS), object existence authority is not required.) Object existence authority is required to create an object that has been named by an authority holder.

**\*OBJMGT:** Object management authority provides the authority to specify the security for the object, move or rename the object, and add members to database files.

**\*OBJOPR:** Object operational authority provides authority to look at the description of an object and use the object, as determined by the data authorities that the user has to the object.

**\*OBJREF:** Object reference authority provides the authority needed to reference an object from another object such that operations on that object may be restricted by the other object. If the user has this authority on a physical file, the user can add referential constraints in which the physical file is the parent. This authority is currently only used for database files.

**\*ADD:** Add authority provides the authority to add entries to an object (for example, job entries to a queue or records to a file).

**\*DLT:** Delete authority allows the user to remove entries from an object, for example, remove messages from a message queue or records from a file.

**\*EXECUTE:** Execute authority provides the authority needed to run a program or to locate an object in a library.

**\*READ:** Read authority provides the authority needed to get the contents of an entry in an object or to run a program.

**\*UPD:** Update authority provides the authority needed to change the entries in an object.

### Example for GRTWSOAUT

```
GRTWSOAUT  WSOTYPE(*TPLWRKARA)  AUTL(KLIST)
```

This command gives authority to the work are template to the users with authority specified for them on the authorization list KLIST.

No error messages.

---

## HLDCMNDEV (Hold Communications Device) Command Description

HLDCMNDEV Command syntax diagram

### Purpose

The Hold Communications Device (HLDCMNDEV) command allows the operator to hold communication through the specified device description in two ways: controlled or immediate. The controlled option allows any program using the communications device to continue to do input/output operations, but no new uses of the device are started. The immediate option stops a communications device immediately, and a permanent input/output error is sent to the program. Communications are restarted with the Release Communications Device (RLSCMNDEV) command or by varying the device off and then on (Vary Configuration (VRYCFG) command).

**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 Parameter

**DEV** Specifies the name of the device whose communications are held. Devices whose communications are held are:

#### DEV Value

##### Device

**3180** Display station

**3277** Display station

**3278** Display station



**3279** Display station  
**3287** Printer (work station)  
**5219** Printer (work station)  
**5224** Printer (work station)  
**5225** Printer (work station)  
**5251** Display station  
**5252** Display station  
**5256** Printer (work station)  
**5291** Display station  
**5292** Display station  
**PLU1** Primary logical unit, type 1 (for SNA)  
**BSC** Binary synchronous device (Base and RJE)  
**B SCT** This iSeries 400 as a BSC multipoint tributary station  
**APPC** Logical unit in advanced program-to-program communications (APPC) network

Specify the name of the device (as specified in the device description) whose communications capability is suspended.

### Optional Parameter

#### OPTION

Specifies the manner in which communications with this device is held.

**\*CNTRL D:** The specified device is not capable of communications at the next OPEN or ACQUIRE operation.

**\*IMMED:** The specified device is not capable of communications at the next READ, WRITE, OPEN, or ACQUIRE operation.

### Example for HLDCMNDEV

```
HLDCMNDEV DEV(WSPR05)
```

This command holds the communications capability of the device WSPR05 at the time of the next OPEN or ACQUIRE operation.

### Error messages for HLDCMNDEV

#### \*ESCAPE Messages

##### CPF5920

Device &1 varied off or in diagnostic mode.

##### CPF5921

Device &1 not a communications device.

##### CPF5922

Device &1 already held with option \*IMMED.

##### CPF5935

Error occurred during command processing.

**CPF5984**

Not authorized to perform function.

**CPF9814**

Device &1 not found.

**CPF9825**

Not authorized to device &1.

---

## HLDDSTQ (Hold Distribution Queue) Command Description

HLDDSTQ Command syntax diagram

### Purpose

The Hold Distribution Queue (HLDDSTQ) command prevents a distribution queue from being sent.

Distribution queue names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

### Restrictions:

1. This command is shipped with public \*EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to use the command.
2. Messages that report errors about distribution queues may display or print different characters than the user entered for the distribution queue name because of internal system transformations. Similarly (depending on the language used for the work station), the internal value for a distribution queue name may differ from the characters shown on the Work with Distribution Queue (WRKDSTQ) command. An error may be reported if the character-string value specified for the DSTQ parameter does not match the rules for an internal distribution queue value or if it does not match the internal value for any defined distribution queue (ignoring case differences).

### Required Parameters

**DSTQ** Specifies the name of the distribution queue being held. Both normal and high priority portions of the specified distribution queue are shown or printed. The queue specified must have been previously configured. See the Configure Distribution Services (CFGDSTSRV) command or the Add Distribution Queue (ADDSTQ) command.

**PTY** Specifies whether the normal priority or high priority portion of the specified queue is held.

**\*NORMAL:** Holds the normal priority queue, which is for distributions with a service level of data low.

**\*HIGH:** Holds the high priority queue, which is for distributions with a service level of fast, status, or data high.

### Examples for HLDDSTQ

#### Example 1: Holding the Normal Priority Portion of a Queue

```
HLDDSTQ DSTQ(CHICAGO) PTY(*NORMAL)
```

This command holds the normal priority portion of the CHICAGO distribution queue.

#### Example 2: Holding the High Priority Portion of a Queue

```
HLDDSTQ DSTQ(ATLANTA) PTY(*HIGH)
```

This command holds the high priority portion of the ATLANTA distribution queue.

## Error messages for HLDDSTQ

### \*ESCAPE Messages

#### CPF8802

Distribution queue &1 was not found.

#### CPF8805

Special value for System name/Group not permitted or not used correctly.

#### CPF8806

Value &1 not valid for system name or system group.

#### CPF881C

High priority queue not allowed for \*SVDS distribution queue &1

#### CPF8812

Error occurred while processing distribution queues.

#### CPF8816

QSNADS communications subsystem is not active.

#### CPF8817

Distribution queue is held.

#### 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.

---

## HLDJOB (Hold Job) Command Description

HLDJOB Command syntax diagram

### Purpose

The Hold Job (HLDJOB) command makes a job ineligible for processing by the system. The job is held until it is:


- Released by the Release Job (RLSJOB) command
- Cleared by the Clear Job Queue (CLRJOBQ) command
- Ended by the End Job (ENDJOB) command
- Ended (while the job is active) by the End Subsystem (ENDSBS) command, the End System (ENDSYS) command, or the Power Down System (PWRDWNSYS) command

If the job is not run before the OS/400 system is ended, the queue can be cleared (and the job ended) when the OS/400 system is started again. The specified job to be held can be on the job queue or on the output queues, or it can be active in a subsystem. Holding a job causes all threads within the job to be held. This command also specifies whether the job's spooled files are held.

### Note:

If you use this command to hold a job that has exclusive access to any resources on the system, these resources are not available to other jobs. Other jobs which require access to those resources will either fail or wait indefinitely.

**Restriction:** 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 held, 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 qualified name of the job being held. If no job qualifier is given, all of the jobs currently in the system are searched for the job name. If more than one of the specified names are found, a qualified job name must be specified.

A job identifier is a special value or 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 being held.

*user-name:* Specify the name of the user of the job being held.



*job-number:* Specify the number of the job being held.

### Optional Parameters

#### SPLFILE

Specifies whether spooled files created by the job being held are also held.

**\*NO:** The spooled files produced by the job are not held.

**\*YES:** The spooled files produced by the job are also held.  If the spooled file action (SPLFACN) job attribute is \*DETACH and the job is ended while the spooled files are held, the spooled files cannot be released using the Release Job (RLSJOB) command. To release spooled files after the job has been removed from the system, use the Release Spooled File (RLSSPLF) command. 

#### DUPJOB OPT

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 HLDJOB

#### Example 1: Making a Job Ineligible for Processing

```
HLDJOB JOB(PAYROLL) SPLFILE(*YES)
```

This command makes the job named PAYROLL ineligible for processing. All spooled files for this job are also held.

#### Example 2: Holding a Job that has a Duplicate Name

```
HLDJOB JOB(DEPTXYZ/PAYROLL)
```

This command holds the job named PAYROLL submitted by a user operating under the user profile DEPTXYZ. The qualified form of the job name is used when jobs with duplicate names exist in the system. Spooled files are not held.

## **Error messages for HLDJOB**

### **\*ESCAPE Messages**

#### **CPF1E52**

Not authorized to hold job &1.

#### **CPF1E53**

Job &1 has ended and cannot be held.

#### **CPF1E54**

Job &1 cannot be held.

#### **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.

#### **CPF1345**

Cannot hold job &3/&2/&1.

#### **CPF1346**

Job &3/&2/&1 already held.

#### **CPF1347**

Cannot hold job &3/&2/&1.

#### **CPF1348**

Job &3/&2/&1 held but unable to hold its files.

#### **CPF1350**

SPLFILE(\*NO) specified but job &3/&2/&1 on OUTQ.

#### **CPF1351**

Function check occurred in subsystem for job &3/&2/&1.

#### **CPF1352**

Function not done. &3/&2/&1 in transition condition.

#### **CPF1378**

Job &3/&2/&1 not held at current time.

---

## HLDJOBQ (Hold Job Queue) Command Description

HLDJOBQ Command syntax diagram

### Purpose

The Hold Job Queue (HLDJOBQ) command prevents the processing of all jobs currently waiting on the job queue and all entries that are added to the queue after the command is issued. The HLDJOBQ command has no effect on jobs that are running. Additional jobs can be placed on the job queue while it is being held, but they are not processed. The jobs are held until a Release Job Queue (RLSJOBQ) command is issued. When a job queue is being held, the jobs can be cleared with the Clear Job Queue (CLRJOBQ) command or a specific job can be canceled by the ENDJOB command.

**Restriction:** The QLPINSTALL job queue cannot be held.

### Required Parameter

**JOBQ** Specifies the qualified name of the job queue that has its current and future jobs withheld from further processing.

The name of the job queue 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.

*job-queue-name:* Specify the name of the job queue being held

### Example for HLDJOBQ

```
HLDJOBQ JOBQ(QBATCH)
```

This command prevents the processing of the jobs currently on the QBATCH job queue and any jobs added to the queue. They are held until the queue is released or cleared. Individual jobs can also be ended with the ENDJOB command, which removes the job from the job queue.

### Error messages for HLDJOBQ

#### \*ESCAPE Messages

##### CPF2207

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

##### CPF2240

User &7 not authorized to use \*&5 &6/&4.

##### CPF3307

Job queue &1 in &2 not found.

##### CPF3330

Necessary resource not available.

---

## HLDJOBSCDE (Hold Job Schedule Entry) Command Description

HLDJOBSCDE Command syntax diagram

### Purpose

The Hold Job Schedule Entry (HLDJOBSCDE) command holds an entry, entries, or generic entries in the job schedule. Each job schedule entry contains the information needed to automatically submit a batch job one time, or at regularly scheduled intervals.

If you hold a job schedule entry:

- The entry is held until it is released using the Release Job Schedule Entry (RLSJOBSCDE) or Work with Job Schedule Entries (WRKJOBSCDE) command.
- A job is not submitted when the entry is released, even if the date and time at which it was scheduled to be submitted passed while the entry was held.

**Restriction:** To hold entries, you must have \*JOBCTL special authority; otherwise you can hold only those entries that you added.

### Required Parameter

**JOB** Specifies the name of the job schedule entry.

**\*ALL:** All of the job schedule entries for which you have authority are held. If JOB(\*ALL) is specified, ENRYNBR(\*ALL) must also be specified.

*job-name:* Specify the name of the job schedule entry.

*generic\*-job-name:* Specify a generic name. 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. If a generic name is specified, ENRYNBR(\*ALL) must also be specified.

### Optional Parameter

#### ENRYNBR

Specifies the number of the job schedule entry you want to hold. The message sent when an entry is successfully added contains the entry number. You can also determine the entry number by using the Work with Job Schedule Entries (WRKJOBSCDE) command. Press F11 from the Work with Job Schedule Entries display to show the entry numbers of the selected entries.

**\*ONLY:** One entry in the job schedule has the job name specified on the JOB parameter. If \*ONLY is specified and more than one entry has the specified job name, no entries are held and a message is sent.

**\*ALL:** All entries with the specified job name are held.

*entry-number:* Specify the number of the job schedule entry you want to hold.

### Examples for HLDJOBSCDE

#### Example 1: Holding a Job Schedule Entry

```
HLDJOBSCDE JOB(CLEANUP)
```

This command holds the job schedule entry with the job name CLEANUP.

**Example 2: Holding All Job Schedule Entries**

```
HLDJOBSCDE JOB(*ALL) ENRYNBR(*ALL)
```

This command holds all entries in the job schedule.

**Example 3: Holding an Individual Job Schedule Entry**

```
HLDJOBSCDE JOB(PAYROLL) ENRYNBR(*ONLY)
```

This command holds the entry PAYROLL in the job schedule.

**Example 4: Holding a Generic Job Schedule Entry**

```
HLDJOBSCDE JOB(PAY*) ENRYNBR(*ALL)
```

This command holds all entries in the job schedule that have the prefix PAY in their names.

**Error messages for HLDJOBSCDE**

**\*ESCAPE Messages**

**CPF1628**

Job schedule entry &3 number &4 not found.

**CPF1629**

Not authorized to job schedule &1.

**CPF1630**

Not authorized to job schedule entry &3 number &4.

**CPF1632**

Job schedule entry &3 number &4 damaged.

**CPF1636**

More than one entry with specified entry job name found.

**CPF1637**

Job schedule &1 in library &2 in use.

**CPF1638**

Job schedule entry &3 number &4 in use.

**CPF1640**

Job schedule &1 in library &2 does not exist.

**CPF1641**

Job schedule &1 in library &2 damaged.

**CPF1645**

No job schedule entries found for specified name.

**CPF1646**

Entry number must be \*ALL when generic name specified.

**CPF1647**

&3 entries successfully held, &4 entries not held.

**CPF1649**

Entry number must be \*ALL.



---

## HLDJOBJS (Hold Job Using Job Scheduler) Command Description

**Note:** To use this command, you must have the 5722-JS1 (Job Scheduler for iSeries) licensed program installed.

HLDJOBJS Command syntax diagram

### Purpose

The Hold Job using Job Scheduler (HLDJOBJS) command allows you to hold a job that you specify.

**Note:** When referring to a job in this command, we are referring to an entry in Job Scheduler. An **entry** in Job Scheduler is a user-defined name for commands or programs that you want to process at scheduled times and dates. Job Scheduler jobs (entries) are not OS/400 objects.

When you press Enter, a message is displayed confirming that the job you selected has been held. The job is held until the Release Job using Job Scheduler (RLSJOBJS) command is processed for the job or Option 6 (Release/Reset) is selected for the job on the Work with Jobs display.

### Required Parameter

**JOB** Specifies the name of the job that you want to hold.

You must specify a job and optionally can specify a group to which the job belongs and the associated sequence number of the job.

#### Element 1: Job

*job-name:* Specify the name of the job that you want to hold.

#### Element 2: Group

**\*NONE:** The job is not a member of a group.

*group-name:* Specify the name of the group to which the job belongs.

#### Element 3: Group sequence

**\*NONE:** The job does not have a sequence number.

*group-sequence-number:* Specify the sequence number of the job in the group.

### Example for HLDJOBJS

#### Example 1: Holding a Job

```
HLDJOBJS JOB(JOB02)
```

In the example, JOB02 is held.

#### Error messages for HLDJOBJS

None

---

## HLDOUTQ (Hold Output Queue) Command Description

HLDOUTQ Command syntax diagram

### Purpose

The Hold Output Queue (HLDOUTQ) command prevents all currently waiting spooled files, and all spooled files that are added to the output queue after the command is issued, from being processed by a spooling writer. The HLDOUTQ command has no effect on jobs that are running and adding spooled files to the queue or on the spooled file being produced by a spooling writer at the time the command is issued. When the spooling writer completes all copies of the current output file, it cannot begin the output for any other files until the queue is released.

The spooled files are held until a Release Output Queue (RLSOUTQ) command is issued. Otherwise, the output queue can be cleared with the Clear Output Queue (CLROUTQ) command or a specific file can be deleted by the DLTSPFL command.

### Required Parameter

**OUTQ** Specifies the qualified name of the output queue.

The name of the output queue 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.

*output-queue-name:* Specify the name of the output queue.

### Example for HLDOUTQ

```
HLDOUTQ OUTQ(QPRINT)
```

This command prevents the processing of the spooled files on the QPRINT queue and any spooled files added to the queue. They are held until the queue is released or cleared. A specific job (with its spooled files) can also be ended with the ENDJOB command, which removes the spooled files from the output queue.

### Error messages for HLDOUTQ

#### \*ESCAPE Messages

##### **CPF2207**

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

##### **CPF3330**

Necessary resource not available.

##### **CPF3357**

Output queue &1 in library &2 not found.

##### **CPF3426**

Output queue &1 in library &2 already held.



---

## HLDPTF (Hold Program Temporary Fix) Command Description

**Note:** To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

HLDPTF Command syntax diagram

### Purpose

The Hold Program Temporary Fix (HLDPTF) command holds a program temporary fix (PTF) save file (SAVF). When a PTF SAVF is put on hold, it is not automatically sent to a service requester by way of electronic customer support.

### Required Parameter

**PTF** Specifies which PTF to hold.

### Optional Parameters

#### LICPGM

Specifies the product identifier (ID) of the PTF being held.

**\*ALL:** All products are searched for the PTF being held.

*product-ID:* Specify the product ID of the PTF being held.

**RLS** Specifies the release level of the PTF to be held.

**\*ONLY:** The PTF exists for only one release level of the product.

*version-release-modification:* Specify the release level of the PTF being held.

### Example for HLDPTF

```
HLDPTF PTF(SI12345)
```

This command holds the SAVFs of the PTF named SI12345. Holding SI12345 prevents it from being automatically distributed to a service requester.

### Error messages for HLDPTF

#### \*ESCAPE Messages

##### SMU1430

Duplicate PTF found.



---

## HLDRDR (Hold Reader) Command Description

HLDRDR Command syntax diagram

### Purpose

The Hold Reader (HLDRDR) command immediately stops the activity of the specified spooling reader. The reader is not ended, and its associated input device is not made available to the system. The reader remains inactive until a Release Reader (RLSRDR) or End Reader (ENDRDR) command is issued. Data is not lost when the reader is held.

### Required Parameter

**RDR** Specifies the name of the spooling reader being held.

**Example for HLDRDR**

HLDRDR RDR(QDKT)

This command causes the diskette reader QDKT to immediately stop reading data. To release the reader, so that it can continue to read data, a Release Reader (RLSRDR) command must be entered. If the End Reader (ENDRDR) command is used, the reader is stopped and the job that was being read in is lost because no job entry was added to the job queue.

**Error messages for HLDRDR**

**\*ESCAPE Messages**

**CPF1E52**

Not authorized to hold job &1.

**CPF1E53**

Job &1 has ended and cannot be held.

**CPF1E54**

Job &1 cannot be held.

**CPF1317**

No response from subsystem for job &3/&2/&1.

**CPF1340**

Job control function not performed.

**CPF1345**

Cannot hold job &3/&2/&1.

**CPF1347**

Cannot hold job &3/&2/&1.

**CPF1350**

SPLFILE(\*NO) specified but job &3/&2/&1 on OUTQ.

**CPF1351**

Function check occurred in subsystem for job &3/&2/&1.

**CPF1352**

Function not done. &3/&2/&1 in transition condition.

**CPF1378**

Job &3/&2/&1 not held at current time.

**CPF3312**

Reader &1 neither active nor on job queue.

**CPF3330**

Necessary resource not available.

**CPF3333**

Reader &3/&2/&1 already held.

**CPF3490**

Not authorized to specified reader.

---

## HLDSPLF (Hold Spooled File) Command Description

HLDSPLF Command syntax diagram

### Purpose

The Hold Spooled File (HLDSPLF) command stops the specified spooled file from additional processing by a spooling writer. If the file is being produced on an output device, the writer stops processing that file and gets the next file to be processed. When the file is released and again selected for output, it is again processed starting at the beginning. If several copies are being produced for the file when it is held, the incomplete copy is again produced from the beginning along with the remaining copies that follow it.

If the specified file is still receiving records from a program that is running when the file is held, the program that is running is unaware that the file is on hold. Also, if the held file is part of a job that produces other spooled files, they can be processed before the held file is released. The held file remains on the output queue and it appears to be the only file produced by the job.

The file is held until one of the following commands is entered: the RLSSPLF (Release Spooled File), the DLTSPLF (Delete Spooled File), the ENDJOB (End Job) with keyword SPLFILE(\*YES) specified, or the CLROUTQ (Clear Output Queue) command. If the file is released before the writer begins producing it, the file is produced in its normal order within the group of files for the job. Production of a job's output is not delayed because some of its files are being held.

### Required Parameter

**FILE** Specifies the name of the spooled file to be held.

**\*SELECT:** All spooled files that meet the selection requirements specified in the SELECT keyword are held. This value is mutually exclusive with the [»](#) JOB, SPLNBR, JOBSYNAME, and CRTDATE parameters. Specifying \*SELECT causes the JOB, SPLNBR, JOBSYNAME, and CRTDATE parameters to be ignored. [«](#)

*spooled-file-name:* Specify the name of the spooled file to be held.

### Optional Parameters

**JOB** Specifies the name of the job that created the file being held. If no job qualifier is given, all of the jobs currently in the system are searched for the simple job name.

A job identifier is a special value or 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.

**\***: The job that issued this HLDSPLF command is the job that produced the file being held.

*job-name:* Specify the name of the job that created the file being held.

*user-name:* Specify the name of the user of the job that created the file being held.

*job-number:* Specify the number of the job that created the file being held.

### SPLNBR

Specifies the number of the spooled file to be held that was created by the specified job. More information on this parameter is in commonly used parameters.

**\*ONLY:** One spooled file from the job has the specified file name. The number of the spooled file is not necessary. If \*ONLY is specified and more than one spooled file has the specified file name, a message is sent.

**\*LAST:** The spooled file with the highest number and the specified file name is used.

» **\*ANY:** The spooled file number is not used to determine which spooled file is used. Use this value when the job system name parameter or the spooled file creation date and time parameter is to take precedence over the spooled file number when selecting a spooled file. «

*spooled-file-number:* Specify the number of the spooled file that has the specified file name to be held.

## » JOBSYSNAME

Specifies the name of the system where the job that created the spooled file (JOB parameter) ran. This parameter is considered after the job name, user name, job number, spooled file name, and spooled file number parameter requirements have been met.

**\*ONLY:** There is one spooled file with the specified job name, user name, job number, spooled file name, spooled file number, and spooled file creation date and time.

**\*CURRENT:** The spooled file created on the current system with the specified job name, user name, job number, spooled file name, spooled file number, and creation date and time is used.

**\*ANY:** The job system name is not used to determine which spooled file is used. Use this value when the spooled file creation date and time parameter is to take precedence over the job system name when selecting a spooled file.

*system name:* Specify the name of the system where the job that created the spooled file ran.

## CRTDATE

Specifies the date and time the spooled file was created. This parameter is considered after the job name, user name, job number, spooled file name, spooled file number, and job system name parameter requirements have been met.

**\*ONLY:** There is one spooled file with the specified job name, user name, job number, spooled file name, spooled file number, and job system name.

**\*LAST:** The spooled file with the latest creation date and time of the specified job name, user name, job number, spooled file name, spooled file number, and job system name is used.

### Element 1: Date spooled file was created

*date:* Specify the date the spooled file was created.

### Element 2: Time spooled file was created

**\*ONLY:** There is one spooled file with the specified job name, user name, job number, spooled file name, spooled file number, job system name, and spooled file creation date.

**\*LAST:** The spooled file with the latest creation time of the specified job name, user name, job number, spooled file name, spooled file number, job system name, and spooled file creation date is used.

*time:* Specify the time the spooled file was created. «

## SELECT

Specifies which group of files are selected to be held. Files can be selected based on user, device, form type, and user data. Only files that meet each of the requirements are selected.

### Element 1: User Values

**\*CURRENT:** Only files created by the user running this command are held.

**\*ALL:** Files created by all users are held.

*user-name:* Specify the names of files to be held that were created by the specified user.

### Element 2: Device Values

**\*ALL:** Files queued for any device or on any object queue are held.

**\*OUTQ:** All files that are not queued for a device are held. These files are on output queues that are not associated with printers.

*device-name:* Specify the names of files to be held that are queued for the specified device.

### Element 3: Form Type Values

**\*ALL:** Files for all form types are held.

**\*STD:** Only files that specify the standard form type are selected.

*form-type:* Specify the form type of the files to be held.

### Element 4: User Data Values

**\*ALL:** Files with any user data tag specified are held.

*user-data:* Specify the user data tags of files to be held.

## OPTION

Specifies which option to use when holding a spooled file. This parameter allows the user to choose when to hold a spooled file. If the spooled file being held is not currently being processed by a spool writer, this parameter is not valid. If the spooled file is being processed by a printer writer, then printing stops and a form feed is done to prepare the printer for the next spooled file to be printed.

**\*IMMED:** The spooled file is held immediately.

**\*PAGEEND:** The spooled file is held at a page boundary.

## Examples for HLDSPLF

### Example 1: Holding a File Created by Another Job

```
HLDSPLF FILE(SHIPITEMS) JOB(00009/JONES/ORDER)
```

This command withholds the spooled file SHIPITEMS, created by the job ORDER, from additional processing.

### Example 2: Holding a File at a Page Boundary

```
HLDSPLF FILE(QPJOBLOG) OPTION(*PAGEEND)
```

This command holds the spooled file QPJOBLOG at a page boundary.

### Example 3: Holding a File Immediately

```
HLDSPLF FILE(QPJOBLOG) OPTION(*IMMED)
```

This command holds the spooled file QPJOBLOG immediately. Holding a spooled file by specifying this option causes the CHGSPLFA command RESTART(\*NEXT) to be inaccurate if the spooled file is currently being processed by a spool writer.

## Error messages for HLDSPLF

### \*ESCAPE Messages

**CPF33D0**

Printer &1 does not exist.

**CPF33D1**

User &1 does not exist.

**CPF3303**

File &1 not found in job &5/&4/&3.

**CPF3309**

No files named &1 are active.

**CPF3330**

Necessary resource not available.

**CPF3337**

File &1 number &2 already held or saved.

**CPF3340**

More than one file with specified name found in job &5/&4/&3.

**CPF3342**

Job &5/&4/&3 not found.

**CPF3343**

Duplicate job names found.

**CPF3344**

File &1 number &2 no longer in the system.

**CPF3357**

Output queue &1 in library &2 not found.

**CPF34A4**

File &1 number &2 not held or deleted.

**CPF3492**

Not authorized to spooled file.



---

## HLDSBMCRQA (Hold Submitted Change Request Activity) Command Description

**Note:** To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

HLDSBMCRQA Command syntax diagram

### Purpose

The Hold Submitted Change Request Activity (HLDSBMCRQA) command holds one or more change request activities.

### Restrictions:

1. Only activities that have a status of Wait, Scheduled, or Ready can be held.
2. You must be either the submitter of the change request or have \*JOBCTL special authority.

### Required Parameter



**CRQ** Specifies the change request name and the change request sequence number of the activities to be held.

**Element 1: Change Request Name**

*change-request-name:* Specify the change request name of the activities that are held.

**Element 2: Sequence Number**

*sequence-number:* Specify the change request sequence number of the activities to be held.

**Optional Parameters**

**ACTIVITY**

Specifies the name of the activity to hold.

**\*ALL:** Hold all of the specified change request activities.

**\*LAST:** Hold the activity named \*LAST. This is not the last activity added to the change request description object. This is the last activity run after the change request is submitted.

*activity-name:* Specify the name of the activity to hold.

**CPNAME**

Specifies the APPN control point names of the managed systems on which this activity is to be performed.

**Element 1: Network Identifier**

**\*ALL:** All the activities for the change request specified are held regardless of the control point name of the managed system on which the activity is to be performed.

**\*NETATR:** Only network ID activities that match the ones defined in the network attributes for this system are held.

*network-identifier:* Specify a network ID. Only activities for the network ID and the control point name specified are held.

**Element 2: Control Point Name**

**\*ALL:** All the activities for the change request specified are held regardless of the control point name of the managed system on which the activity is to be performed.

**\*NETATR:** Only activities for the control point name that matches the one defined in the network attributes for this system are held.

*control-point-name:* Specify a control point name. Only activities for the network ID and for the control point name specified are held. For NetView Distribution Management Agents, the control point name is the change control client which supports numeric characters (0-9) in the first position of control point names that are valid in other platforms.

**Example for HLDSBMCRQA**

HLDSBMCRQA CRQ(CHG001 456)

This command holds all of the activities for the change request named CHG001 with sequence number 456 for all the nodes.

**Error messages for HLDSBMCRQA**

**\*ESCAPE Messages**

None <<

---

## HLDWTR (Hold Writer) Command Description

HLDWTR Command syntax diagram

### Purpose

The Hold Writer (HLDWTR) command stops the specified writer at the end of a record, at the end of a spooled file, or at the end of a printed page. If multiple copies of a file are produced, the writer can be held at the end of the copy currently being produced. The writer is not ended and the device is not made available to the system. The writer remains inactive until a RLSWTR (Release Writer) or ENDWTR (End Writer) command is issued. Data is not lost when the writer is held.

### Required Parameter

**WTR** Specifies the name of the spooling writer being held. Specify the name of the spooling writer.

### Optional Parameter

#### OPTION

Specifies when the spooling writer should stop producing output.

**\*IMMED:** The writer stops immediately after it has written the last record, in the current block of records, to the output device. Each time the writer finishes producing a block of records on a device, it makes another I/O request to get the next block from the file being spooled to the device. If \*IMMED is specified, the writer stops only after it has written the last record in the block being processed, which (for diskette output) is a complete diskette record being written on diskette.

When \*IMMED is specified for printed output, the writer stops anywhere within or at the end of a print line or at the end of a complete block, which may not be at the end of a line. This is because some data records (which are blocked to improve performance) may be split in two, with the first part of a record at the end of one block and the last part of the record at the beginning of the next block. If only one copy of the file is being produced or if the last copy is being produced, the entry for the file is removed from the output queue when the output is completed.

**\*CNTRLD:** The job is ended in a controlled manner. This allows the program to perform cleanup (end-of-job processing).

**\*PAGEEND:** The writer is held at the end of a page. This value is valid only when the spooling writer is a printer writer.

### Example for HLDWTR

```
HLDWTR WTR(PRINTER) OPTION(*CNTRLD)
```

This command stops the writer named PRINTER at the end of the current file. The writer is held until an RLSWTR (Release Writer) or ENDWTR (End Writer) command is issued.

### Error messages for HLDWTR

#### \*ESCAPE Messages

##### CPF1340

Job control function not performed.

##### CPF3313

Writer &1 not active nor on job queue.

##### CPF3330

Necessary resource not available.

**CPF3331**

Not authorized to control writer &3/&2/&1.

**CPF3332**

Writer &3/&2/&1 already held.

**CPF3334**

Previous hold to writer &3/&2/&1 pending.

**CPF3438**

\*PAGEEND not valid for writer &3/&2/&1.

## IF (If) Command Description

IF Command syntax diagram

### Purpose

The If (IF) command evaluates a logical expression and conditionally processes CL program commands according to the evaluation of the expression. If the logical expression is true (a logical 1), the command (or the group of commands in a Do group) specified in the THEN parameter is processed, and the ELSE command with its associated command or Do group is not processed. If the logical expression is false (a logical 0), the command specified in the THEN parameter is not processed and control passes to the next command. If that command is an ELSE command, the command or Do group specified in that command is processed. If the ELSE command is not specified, control passes to the next command.

When a DO command is specified, either in the THEN parameter of the IF command or in the CMD parameter of the ELSE command, the Do group is bypassed if the result of the expression is not the one needed for the group being processed. That is, control passes to the command that follows the ENDDO command associated with that DO.

When the command or Do group specified by the THEN parameter or the ELSE command is completed (and no GOTO command has been processed), control passes to the next command following the command or Do group specified by the ELSE command. If a GOTO command is processed, control is passed to the command with the label specified by the GOTO command, and processing continues from that command.

The following command sequence shows this flow. In this example, &TESTSW is a logical variable.

```

IF &TESTSW DO
  Group A (group of CL commands)
  *
  *
  ENDDO

ELSE DO
  Group B (group of CL commands)
  *
  *
  ENDDO

Group C (continued CL commands)
*
*
```

The IF command tests the logical variable &TESTSW. If a true condition exists (&TESTSW contains a value of '1'), the commands in Group A are processed, then control passes to the commands in Group C. If a false condition exists (&TESTW contains a value of 0), the commands in group A are bypassed, the commands in Group B are processed, then control passes to the commands in Group C.

### Restrictions:

1. The IF command is valid only in CL programs.
2. Up to ten levels of nested IF and ELSE commands are allowed.

### Required Parameter

**COND** Specifies the logical expression that is evaluated to determine a condition in the program and what is done next. See logical expressions for more information. Note that variables, constants, and the %SUBSTRING, %SWITCH, and %BINARY built-in functions can be used within the expression.

### Optional Parameter

**THEN** Specifies the commands (in a Do group) that are processed if the result of evaluating the expression is true. After the command or Do group is processed, control is passed to the next command *after* the ELSE command associated with this IF command. If the result is true, the ELSE command associated with the IF command is not processed. If the command specified in this parameter is a DO command, all commands within the Do group are considered to be the command specified by the parameter.

If the command specified by the THEN keyword is not coded on the same line when the keyword is coded, the THEN keyword must be immediately followed (on the same line) either by the left parenthesis or by a plus sign (+) or a minus sign (-) to show continuation. (A blank cannot immediately follow any keyword.) The command and the right parenthesis can then be coded on the next line. For example:

```
IF COND(&A *EQ &B) THEN(      &#43;  
    GOTO C)
```

If any part of the command specified by the THEN parameter 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 within the Do group) is within the parentheses. For example:

```
IF COND(&A *EQ &B) THEN(DO)  
    CMD1  
    CMD2  
    *  
    *  
ENDDO
```

If no command is specified on the THEN parameter (a null THEN) and the ELSE command immediately follows it, the ELSE is processed if the IF expression is false and it is skipped if the expression is true.

Any CL command can be specified on the THEN parameter, except the following commands:

```
ELSE  
PGM, ENDPGM  
ENDDO  
MONMSG  
DCL, DCLF
```

The command can be another IF, unless there are already ten levels of nested IF and ELSE commands.

### Examples for IF

```

IF COND(&A *EQ &B) THEN(GOTO X)
IF (&A *EQ &B) THEN(GOTO X)
IF (&A *EQ &B) (GOTO X)

IF COND(&A *EQ &B) THEN(GOTO X)

```

The examples above show a number of different ways the IF command can be specified to test a condition and branch to a label. In each of these examples, if &A equals &B, control passes to a CL command that has a label named X.

```
IF COND(&TESTSW) THEN(CHGVAR VAR(&A) VALUE(23))
```

If &TESTSW has a logical value of 1 (true), the CHGVAR command is processed to set &A to decimal 23; if &TESTSW has a logical value of 0 (not true), the Change Variable (CHGVAR) command is not processed.

```
IF ((&ALPHA *EQ &BETA) *AND *NOT &GAMMA) THEN(RETURN)
```

If the value of &ALPHA equals the value of &BETA and if &GAMMA is a logical 0, then return to the program that called this CL program.

```
IF &LOG1 THEN(IF (&A *GT 10) THEN(GOTO X))
  ELSE(GOTO Y)
ELSE DO
  *
  * (group of CL commands)
ENDDO
```

This is an example of nested IF commands. If &LOG1 has a logical value of 1 (true) and if &A is greater than 10, a branch is made to label X. If &LOG1 has a logical value of 1 and &A is *not* greater than 10, a branch is made to label Y. If &LOG1 has a logical value of 0 (false), &A is not compared to 10. Instead, the DO group of the second ELSE command is processed.

```
IF &TEST THEN(
  DO
    CHGVAR &A (&A &#43; 1)
    GOTO X
  ENDDO
ELSE
  DO
    CHGVAR &B (&B &#43; 1)
    CALL EXTPGM (&B)
  ENDDO
```

This example shows how the THEN parameter can be continued on the next line. If &TEST has a logical value of 1, the Do group specified in the THEN parameter is processed. Otherwise, the Do group specified by the ELSE command is processed.

```
IF (&A *EQ YES)
DO
  CHGVAR &B 1
  CHGVAR &C 'Z'
ENDDO
```

This example shows a Do group as the THEN parameter. The two Change Variable (CHGVAR) commands are processed if, in the relational expression, &A is equal to YES.

```
IF %SWITCH(10XXX10) THEN(GOTO X)
```

This example shows how the built-in function %SWITCH is used to test the eight job switches in a job. Refer to the end of for a complete description of %SWITCH. In this example, job switches 1, 2, 7, and 8 are tested for the values indicated in the 8-character mask. If switches 1 and 7 contain 1s and switches 2 and 8 contain 0s, then the program branches to the command having the label X. If any of the four switches do not contain the value indicated, the branch does not occur.

## Error messages for IF

## **\*ESCAPE Messages**

### **CPF0816**

%SWITCH mask &1 not valid.

---



## **INZBRM (Initialize BRMS) 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.

INZBRM Command syntax diagram

### **Purpose**

The Initialize BRMS (INZBRM) command performs many types of initialization. These are:

- Initializes all major files as well as establishing default policies and control groups
- Clears and then re-initializes the device file and media library file
- Starts the subsystem for networking in a multi-system environment
- Allows you to reset BRMS and re-initialize all major files as well as establishing default policies and control groups
-  Allows you to re-register all BRMS functional authority elements with the OS/400 registration facility. This option is used during a full system recovery prior to restoring user profiles.
- Updates the auxiliary storage pool (ASP) descriptions.
- Allows you to change the system name for BRMS media information to a new system name when restoring this information to a different system or logical partition. 

### **Notes:**

1. It is important to run the INZBRM command on the system that you are adding with OPTION(\*NETTIME) to assure that all network system times are synchronized.
2. The INZBRM command is used to add a system to a BRMS network group and synchronize the time on the network. The INZBRM command must be processed from the system that you are adding to the network group.
3. All references to system name assume that the system name and system location name are the same and are used interchangeably in the help information. If they are not the same, use the system location name instead of the system name. You can review the setting for the system name and location name by using the Display Network Attributes (DSPNETA) command from any command line.

Systems that are members of the network group share the BRMS media inventory. Additions, changes and removals from the media inventory on any system in the network result in the same changes being made to all systems that are members of the network group.

When a system is added to a network group, the media information common to the network group is copied to the system that you are adding. The system's media information is replaced by the network group's media information. When a system is first added, it is added in an inactive status on an active network group member.

### **Note:**

This is accomplished by adding the system to a network group list using the Change Network Group display which is found in the System Policy menu. When the system name is added to the list, it is shown in inactive status.

The INZBRM OPTION(\*NETSYS) option changes the status from inactive to active and synchronizes media information.

The following shared media information is copied from an active system in a network to replace the media information on the incoming system:

- Media inventory
- Media classes
- Media policies
- Container inventory
- Container classes
- Move policies
- Network systems
- Storage locations
- Duplication cross reference

**Note:**

This command should not be used by control group \*EXIT item processing as results will be unpredictable.

### Example for INZBRM

#### Example 1: Initializing the BRMS Product

```
INZBRM OPTION(*NETSYS) FROMSYS(MBANETID.MBASYSID)
```

In this example the BRMS system issuing the command is being initialized from the MBASYSID.

#### Error messages for INZBRM

None

---

## INZPCS (Initialize Client Access/400) Command Description

INZPCS Command syntax diagram

### Purpose

The Initialize Client Access/400 (INZPCS) command is used to establish an operating environment for Client Access applications. This is done by creating various control documents on the Client Access folders. These control documents include code page mapping tables, keyboard tables, and font files used for displaying information on the PC display.

### Optional Parameters

#### KBDTYPE

Specifies the keyboard type to use.

**\*DFT:** The default keyboard type is used. When the command is first run, the default value comes from the system value QKBDTYPE. When the command is run after the first time, the default takes the value specified in the previous running of the command.

*keyboard-type:* Specify the 3-character keyboard type identifier to use. Values are listed in the Keyboard Mapping table.

**Table 1. Keyboard Mapping**

<b>Language/Country or Region</b>	<b>Identifier</b>	<b>ASCII Device Group</b>
Albania	ALI	
Arabic X/Basic	CLB	D
Austria/Germany	AGB	A, B
Austria/Germany Euro Currency	AGE	
Austria/Germany Multinational	AGI	A, B
Belgium Multinational	BLI	B
Belgium Multinational Euro Currency	BLM	
Brazilian Portuguese	BRB	
Brazilian Portuguese Euro Currency	BRE	
Bulgaria	BGB	
Canadian French	CAB	A, B
Canadian French Multinational	CAI	A, B
Canadian French Multinational Euro Currency	CAM	
Chinese (Simplified)	RCB	
Chinese (Traditional)	TAB	
Croatia	YGI	
Cyrillic	CYB	
Czech Republic	CSB	
Denmark	DMB	B
Denmark Euro Currency	DME	
Denmark Multinational	DMI	B
Estonia	ESB	
Finland/Sweden	FNB	B
Finland/Sweden Euro Currency	FNE	
Finland/Sweden Multinational	FNI	B
France (Azerty)	FAB	A, B
France (Azerty) Euro Currency	FAE	
France (Azerty) Multinational	FAI	A, B
France (Qwerty)	FQB	
France (Qwerty) Multinational	FQI	
Greece <sup>1</sup>	GNB <sup>1</sup>	
Hebrew	NCB	D
Hungary	HNB	
Iceland	ICB	
Iceland Euro Currency	ICE	
Iceland Multinational	ICI	
International	INB	
International Multinational	INI	
Iran (Farsi)	IRB	
Italy	ITB	A, B
Italy Euro Currency	ITE	
Italy Multinational	ITI	A, B
Japan English	JEB	
Japan English Multinational	JEI	
Japan Kanji (for PS/55 and 5295 display stations)	JKB	
Japan Latin Extended	JPB	
Japan United States Basic	JUB	
Japan Katakana (for 5251, 5291, 5292, and 3180 Katakana display stations)	KAB	
Korea	KOB	



Language/Country or Region	Identifier	ASCII Device Group
Latin-2/ROECE	ROB	
Latvia	LVB	
Lithuania	LTB	
FYR Macedonia (Former Yugoslav Republic)	MKB	
Netherlands	NEB	
Netherlands Euro Currency	NEE	
Netherlands Multinational	NEI	
Norway	NWB	B
Norway Euro Currency	NWE	
Norway Multinational	NWI	B
Poland	PLB	
Portugal	PRB	B
Portugal Euro Currency	PRE	
Portugal Multinational	PRI	B
Romania	RMB	
Russia	RUB	
Serbia (Cyrillic)	SQB	
Serbia (Latin)	YGI	
Slovakia	SKB	
Slovenia	YGI	
Spain	SPB	B
Spain Euro Currency	SPE	
Spain Multinational	SPI	B
Spanish Speaking	SSB	B
Spanish Speaking Multinational Euro Currency	SSE	
Spanish Speaking Multinational	SSI	B
Sweden	SWB	B
Sweden Euro Currency	SWE	
Sweden Multinational	SWI	B
Switzerland/French Multinational	SFI	B
Switzerland/French Multinational Euro Currency	SFM	
Switzerland/German Multinational	SGI	B
Switzerland/German Multinational Euro Currency	SGM	
Thai	THB	
Turkey (Qwerty)	TKB	
Turkey (F)	TRB	
Ukraine	UAB	
United Kingdom	UKB	A, B
United Kingdom Euro Currency	UKE	
United Kingdom Multinational	UKI	A, B
United States/Canada	USB	A, B, C
United States/Canada Euro Currency	USE	
United States/Canada Multinational	USI	A, B, C
Urdu	PKB	
Vietnam	VNB	
Languages of the former Yugoslavia	YGI	

**Note:**

- <sup>1</sup> The GNB code is the current identifier for Greece. The GKB code was used prior to V2R1, and continues to be supported, but provides fewer characters than the recommended GNB code.

**Note:**

For example, KBDTYPE(USB) indicates a keyboard using the United States/Canada character set.

**ASCII** Specifies the ASCII code page number to use.

**\*DFT:** The default PC code page number is used. When the command is run for the first time, the default value is 437 for keyboard types USB and USI and 850 for most others. When the command is run after the first time, the default value takes the value specified at the previous running of the command.

*code-page-number:* Specify the ASCII code page number to use.

**EBCDIC**

Specifies the EBCDIC (or host) code page number to use.

**\*DFT:** The default system code page number is used. When the command is first run, the default value comes from the code page portion of the system value QCHRID. When the command is run after the first time, the default takes the value specified in the previous running of the command.

*code-page-number:* Specify the 3-character EBCDIC (or host code) page number to use.

**LANGUAGE**

Specifies the language feature identifier (ID) of the secondary language to be processed.

**\*DFT:** The primary language of Client Access is processed.

*language-feature-code:* Specify the language feature code of the language to process. Values are listed in Table 2 (92).

**Table 2. Language Feature Identifiers**

<b>Language</b>	<b>Identifier</b>
Arabic	2954
Belgium Dutch	2963
Belgium French	2966
Brazilian Portuguese	2980
Canadian French	2981
Croatian	2912
Czech	2975
Danish	2926
Dutch	2923
English	2924
English U/L (DBCS)	2984
English Uppercase	2950
English Uppercase/DBCS)	2938
Farsi	2998
Finnish	2925
French	2928
French-MNCS	2940
German	2929
German-MNCS	2939
Greek	2957
Hebrew	2961
Hungarian	2976
Icelandic	2958
Italian	2932
Italian-MNCS	2942
Japanese (DBCS)	2962

<b>Language</b>	<b>Identifier</b>
Korean (DBCS)	2986
Norwegian	2933
Polish	2978
Portuguese	2922
Portuguese-MNCS	2996
Russian	2979
Slovakian	2994
Slovenian	2911
Spanish	2931
Swedish	2937
Thai	2972
Traditional Chinese (DBCS)	2987
Turkish	2956

### **Example for INZPCS**

INZPCS

This command copies or merges all machine readable information (MRI) into personal computer programs. The default keyboard type used is determined from the system value QKBDTYPE. A default ASCII code page is selected based on the KBDTYPE. This is usually 850 for keyboard types other than USB or USI. The EBCDIC code page number is determined from the system value QCHRID.

### **Error messages for INZPCS**

#### **\*ESCAPE Messages**

##### **IWS16D0**

Initialize Client Access/400 (INZPCS command) failed.

##### **IWS16DD**

Error getting message &1 from message file &2 in library &3.

##### **IWS16E1**

INZPCS command successfully completed.

##### **IWS16E2**

Error retrieving data area &1 in library &2.

##### **IWS16E3**

Error creating data area QINZPCSDA in library QUSRSYS.

##### **IWS16E4**

Error updating data area QINZPCSDA in library QUSRSYS.

##### **IWS16EE**

Failed to delete data area &1 in library &2.

---

## **INZDKT (Initialize Diskette) Command Description**

INZDKT Command syntax diagram

### **Purpose**

The Initialize Diskette (INZDKT) command initializes a diskette for use. This command writes identification information on a diskette and sets the format. Initializing a diskette includes the following:

- Checking for active files that should not be cleared.

- Testing each track for physical defects on the recording surface. A diskette is unusable if more than two defective cylinders are found, if cylinder 0 is defective, or if the track identifier of a defective track cannot be read.
- Formatting each track to a specified sector size (128, 256, 512 or 1024 bytes) and recording density (single density or double density). A diskette's format determines what the diskette may be used for in later processing. This is explained more fully in the FMT and SCTSIZ parameter.
- Defining a single (expired) file covering the entire diskette. The file is identified as DATA.

IBM-supplied diskettes are initialized before they are shipped to a customer. A diskette should be reinitialized when:

- Its format is changed.
- The sequence of the records on the diskette is changed (they can only be sequential).
- A defect has occurred in one or two tracks.
- The diskette has been exposed to a strong magnetic field.

One INZDKT command can initialize only one diskette at a time.

**Note:** When initializing diskettes with labels that are not IBM standard labels, specify CHECK(\*NO).

### Required Parameter

**DEV** Specifies the name of the device in which the diskette being initialized is placed.

### Optional Parameters

#### NEWVOL

Specifies the volume identifier for the diskette being initialized.

**\*NONE:** No volume identifier is specified; only the system date is written in the volume identifier field of the volume label.

*volume-identifier:* Specify up to 6 characters for the volume identifier to identify the diskette being initialized; any combination of alphabetic characters (except \$, #, or @) or numeric characters can be used.

#### NEWOWNID

Specifies the identification of the new diskette owner to write in the volume label. Any combination of alphabetic characters (except \$, #, or @) or numeric characters can be used.

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

*owner-identifier:* Specify up to 14 uppercase alphabetic or numeric characters that identify the new owner of the diskette. Even if the value is enclosed in apostrophes, no lowercase letters, embedded blanks, or special characters can be used. If fewer than 14 characters are specified, the field is left-justified and padded on the right with blanks.

**FMT** Specifies the format to use to initialize the diskette. Either \*DATA, \*DATA2, 1, 2, or 2D can be specified if the diskette will contain data files that are in the basic, H, I, or System/36 environment exchange formats. The basic exchange format has a set of requirements that ensures that a diskette can be exchanged between systems that are capable of using both the IBM diskette types 1 or 2. \*SAVRST (type E general data exchange) must be specified if the diskette is used in the iSeries 400 or System/38 save and restore operations; that is, if their data files will contain saved objects. \*SAVRST is also a valid format for the System/36 save and restore operations, and it is the preferred format to use.

**\*DATA:** A one- or two-sided diskette is formatted with single-density recording.

**1:** A one-sided diskette is formatted with single-density recording.

**2:** A two-sided diskette is formatted with single-density recording.

**2D:** A two-sided diskette is formatted with double-density recording.

**\*DATA2:** A two-sided diskette is formatted with double-density recording (as if 2D is specified).

**\*SAVRST:** A two-sided diskette is formatted with double-density recording. This format is required if the diskette is used in a save and restore operation.

**Notes:**

1. Because \*DATA2, \*SAVRST, and 2D specify that the diskette is used for double-density recording, it is recommended that a type 2D diskette be used, rather than a type 2. A type 2D diskette is made for double-density recording, whereas a type 2 is made for single-density recording and is more prone to media errors if used for double-density recording.
2. If FMT(\*SAVRST) is specified, CODE(\*ASCII) cannot be specified.

**SCTSIZ**

Specifies the number of bytes per sector with which each track is initialized.

**\*STD:** A standard sector size, based on the value of the FMT parameter, is used. Such as the following:

FMT	SCTSIZ (*STD)
*DATA	128
1	128
2	128
2D	256
*DATA2	256
*SAVRST	256

**128:** Each track is initialized with 128 bytes per sector.

**256:** Each track is initialized with 256 bytes per sector.

**512:** Each track is initialized with 512 bytes per sector.

**1024:** Each track is initialized with 1024 bytes per sector.

Table 1 (page 96), at the end of this command description, shows the valid sector sizes.

Format for 8-inch diskette.

Table 2 (page 96), at the end of this command description, shows the format for 5.25-inch diskette

The following chart shows which exchange types can be used for each sector size.

Exchange Type	128	256	512	1024
Basic	X			
H		X <sup>1</sup>		
I	X	X	X	X
SAVRST				X
S/36-S/R	X	X	X	X

<sup>1</sup> H is used only as the exchange type if the diskette is initialized with double-density recording (FMT(\*DATA2) or FMT(2D)).

## CHECK

Specifies whether a check is made for active files (files having an expiration date greater than the system date).

**\*YES:** A check is performed on files whose labels are in cylinder 0 only. File labels in an extended file label area are not checked. If any active files are found, an operator message is sent. The operator can continue initialization (active files are destroyed), or end the initialization of that diskette.

**\*NO:** Diskette initialization proceeds without a check for active files.

**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 volume label is written in EBCDIC and is an IBM standard label; subsequent data must also be written in EBCDIC.

**\*ASCII:** The volume label is written in ASCII and is an IBM standard label in the same format as the EBCDIC label; subsequent data must also be written in ASCII. If FMT(\*SAVRST) is specified, \*ASCII cannot be specified.

**Table 1. Valid Sector Size**

SCTSIZ	*DATA	1	2	2D	*DATA2	*SAVRST
*STD	X	X	X	X	X	X
128	X	X	X			
256	X	X	X	X	X	
512	X	X	X	X	X	
1024				X	X	X

**Table 2. Format for 5.25 Inch Diskette**

SCTSIZ	*DATA	1	2	2D	*DATA2	*SAVRST
*STD				X	X	X
256				X	X	
512				X	X	
1024				X	X	X

## Examples for INZDKT

### Example 1: Initializing a Diskette

```
INZDKT DEV(DKT1) NEWVOL(ORIGIN) NEWOWNID(DEPT504)
```

This command initializes the diskette device DKT1. The diskette is checked for active files (CHECK(\*YES) is assumed). The diskette is formatted for basic data exchange files (FMT(\*DATA) is assumed). The volume label has ORIGIN written in the volume identifier field and DEPT504 written in the owner identifier field.

### Example 2: Initializing a Diskette to the Save and Restore Format

```
INZDKT DEV(DKT2) NEWVOL(SAVE)  
NEWOWNID(DON) FMT(*SAVRST) CHECK(*NO)
```

This command initializes the diskette in device DKT2 to the save and restore format. The diskette is initialized with the NEWVOL parameter of SAVE. The owner identifier field has DON written into it.

## Error messages for INZDKT

## **\*ESCAPE Messages**

### **CPF6156**

Cancel reply received for message &6.

### **CPF6716**

Device &1 not a diskette device.

### **CPF6717**

Initialize diskette ended; previous errors occurred.

### **CPF6718**

Cannot allocate device &1.

### **CPF6757**

Owner identifier &1 contains wrong characters.

### **CPF6758**

Volume identifier &1 contains wrong characters.

### **CPF6779**

Format (FMT) specified for diskette in device &1 not valid.

### **CPF9814**

Device &1 not found.

### **CPF9825**

Not authorized to device &1.

---

## **INZDSTQ (Initialize Distribution Queue) Command Description**

INZDSTQ Command syntax diagram

### **Purpose**

The Initialize Distribution Queue (INZDSTQ) command resets the status of a distribution queue and the entries on the queue. It also optionally clears all distributions on the queue. This command applies to both the normal and high priority sections of the specified queue.

### **Attention:**

Initializing a distribution queue can result in the loss or duplication of distributions in the network, depending on the status of the distributions in transit at the time this command is run.

Initializing a distribution queue includes the following:

- If a SNADS (Systems Network Architecture (SNA) distribution services) sender job is active for the queue, the job is ended. This job cancellation takes effect immediately. Distributions being sent are interrupted.
- If the queue type is a SystemView distribution services (SVDS) queue type and a receiver job is active for this connection, the job is ended. This job cancellation takes effect immediately. All partially received distributions are discarded.
- If the distribution queue is to be cleared, all distributions on the queue are deleted as specified on the CLEAR parameter.
- If the queue is not cleared, the distributions on the queue that do not have "Held" status are set to "Ready". Distributions with a status of "Held" remain held.
- The queue status is set to "Ready" unless the queue is in the "Held" status.
- If the QSNADS system is active, a SNADS sender job is submitted for the queue following the same rules used to start the QSNADS subsystem.

Distribution queue names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

### Restrictions:

1. This command is shipped with public \*EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to use the command.
2. Messages that report errors about distribution queues may display or print different characters than the user entered for the distribution queue name because of internal system transformations. Similarly (depending on the language used for the work station), the internal value for a distribution queue name may differ from the characters shown on the Work with Distribution Queue (WRKDSTQ) command. An error may be reported if the character-string value specified for the DSTQ parameter does not match the rules for an internal distribution queue value or if it does not match the internal value for any defined distribution queue (ignoring case differences).

### Required Parameter

**DSTQ** Specifies the name of the distribution queue to initialize. The queue must be previously configured using the Configure Distribution Services (CFGDSTSRV) or the Add Distribution Queue (ADDSTQ) command.

### Optional Parameter

#### CLEAR

Specifies whether distributions on the queue are deleted.

#### Attention:

Using the \*PURGE value results in the loss of distributions with no trace.

**\*NO:** Distributions on the queue are not deleted.

**\*YES:** Distributions on the queue are deleted. Each deleted distribution is logged and, if the distribution originator requested notification, a notification is sent to the originator or to the report destination specified in the distribution.

#### Note:

System Network Architecture distribution services (SNADS) status distributions and distribution reports are used to report information about a distribution back to the originator. Status report distributions never result in another status report distribution. If a status report distribution is deleted, no notification is sent.

**\*PURGE:** Distributions on the queue are deleted. Deleted distributions are not logged and no notification is sent to the originator or to the report destination specified in the distribution.

### Examples for INZDSTQ

#### Example 1: Initializing a Distribution Queue

```
INZDSTQ DSTQ('SYSTEMA APPN')
```

Connection information is about to be changed for system 'SYSTEMA APPN' by a central site administrator. This command initializes the queue to avoid error conditions that can be encountered by the Change Distribution Queue (CHGDSTQ) command. Distributions on the queue are not deleted.

#### Example 2: Initializing and Clearing a Distribution Queue

```
INZDSTQ DSTQ('ERRORQ') CLEAR(*YES)
```



This command clears the distribution queue ERRORQ that is being used as a repository for distributions that would have resulted in routing errors. Distributions that are deleted are logged, and the originators of the distributions are notified.

### Example 3: Initializing and Purging a Distribution Queue

```
INZDSTQ DSTQ('TESTQ') CLEAR(*PURGE)
```

This command clears the distribution queue TESTQ that is being used for testing a new batch application. Distributions are deleted but not logged, and the originators are not notified.

### Error messages for INZDSTQ

#### \*ESCAPE Messages

##### CPF8802

Distribution queue &1 was not found.

##### CPF8807

Error occurred while using QSNADS journal.

##### CPF8809

Errors detected on SNADS internal queues.

##### CPF8812

Error occurred while processing distribution queues.

##### CPF8849

Queue &1 in use by another distribution services function.

##### CPF9845

Error occurred while opening file &1.

##### CPF9846

Error while processing file &1 in library &2.

---

## INZMEDBRM (Initialize 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.

INZMEDBRM Command syntax diagram

### Purpose

The Initialize Media using BRM (INZMEDBRM) command prepares media for use in the BRMS system. This command is used to initialize a volume with a standard volume label for standard label magnetic volume processing.

### Notes:

1. It is recommended that you use the INZMEDBRM command in place of the OS/400 INZTAP command. To assure the protection of media BRMS disables INZTAP CHECK(\*NO) for users who do not have \*SECOFR, \*SERVICE or \*SAVSYS authority. Unlike users of INZTAP, users of INZMEDBRM do not need these levels of authority in order to use the CHECK(\*NO) option.
2. 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.
3. If you try to initialize a new tape that has never been written on you will receive an OS/400 error. This is because the first step in the INZMEDBRM command is to issue the OS/400 Check Tape (CHKTAP)

command. OS/400 issues a media error message to QSYSOPR if it does not find a volume identifier on the tape. Even though this message is issued, the tape is successfully initialized, unless there are other error conditions encountered.

### Example for INZMEDBRM

#### Example 1: Initializing a Volume

```
INZMEDBRM DEV(TAP06) NEWVOL(T00004) MEDCLS(QIC1000)
```

In this example the volume T00004 is being initialized using device TAP06. The volume is assigned a media class of QIC1000 and initialized using the density specified by the QIC1000 media class.

### Error messages for INZMEDBRM

None

---

## INZOPT (Initialize Optical) Command Description

INZOPT Command syntax diagram

### Purpose

The Initialize Optical (INZOPT) command initializes an optical volume. Depending on the type of optical volume being initialized this operation may take 30 minutes or more to complete. When an existing optical volume is initialized a second time, all existing information is lost.

Media format is determined by the media type. If the media type is \*DVD-RAM, the new media format will be \*UDF (Universal Disk Format). If the media type is \*ERASE or \*WORM, the new media format will be \*HPOFS (High Performance Optical File System).

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

### Optional Parameters


**VOL** Specifies the volume identifier of the optical volume being initialized or re-initialized.

**\*MOUNTED:** The volume mounted on the specified device (DEV parameter) will be initialized.

*volume-identifier:* Specify the identifier of the optical volume to initialize.

### NEWVOL

Specifies the identifier of the optical volume after it is initialized. The identifier must contain only alphabetic characters (A through Z), numeric characters (0 through 9), a hyphen (-), or a period (.). The first character must be alphabetic or numeric and the identifier cannot contain blanks. More

information about optical volume names can be found in the Optical Support  book.

**\*VOL:** The new volume identifier is the same as the old volume identifier.

*new-volume-identifier:* Specify the new volume identifier.

**DEV** Specifies the name of an optical device which contains the volume to be initialized. This parameter is required when VOL(\*MOUNTED) is specified. The device cannot be an optical media library device.

*optical-device:* The name of the optical device containing the volume which will be initialized.

## THRESHOLD

Specifies the percentage of space on the volume to use until the volume is considered full. This field is only used for volumes in a media library device with a target media format of \*HPOFS. For volumes in a DVD-RAM device, or volumes with a target media format of \*UDF, this field is ignored and the threshold will default to 100 percent.

**Note:** If TYPE(\*BACKUP) is specified, this parameter is ignored and the volume-full-threshold is set to 99 percent.

**\*CALC:** The system will calculate the percentage of the volume to use based on media format and volume type.

- For a media format of \*HPOFS and a volume type of \*PRIMARY the threshold will be 95 percent.
- For a media format of \*HPOFS and a volume type of \*BACKUP the threshold will be 99 percent.
- For a media format of \*UDF the threshold will be 100 percent.

*volume-full-threshold:* Specify the volume threshold percentage. Valid values range from 1 through 100.

**Note:** If the volume type is \*BACKUP, this parameter is ignored and the volume-full-threshold is set to 99 percent.

If the media format is \*UDF, this parameter is ignored and the volume-full-threshold is set to 100 percent.

## CHECK

Specifies whether the system checks to see if the optical volume is already initialized.

**\*YES:** The system checks to see if the optical volume is initialized.

**Note:** If the optical volume is initialized, the operation is ended and an error message is sent.

**\*NO:** The system does not check to see if the optical volume is initialized.

## » ENDOPT

Specifies whether the media is unloaded from the device after the initialize completes.

**Note:** This parameter is ignored if the media is in an optical library device.

**\*LEAVE:** When the initialize completes the media is left in the device.

**\*UNLOAD:** When the initialize completes the media is unloaded from the device. «

## CLEAR

Specifies whether or not existing data on the volume will be cleared during the initialize process. This parameter only applies when the volume media type is \*DVD-RAM.

**Note:**

If the volume media type is \*WORM the volume is never cleared regardless of the parameter setting.

If the volume media type is \*ERASE the volume is always cleared regardless of the parameter setting.

**\*NO:** >> The volume is not cleared. <<

**\*Yes:** >> The volume is cleared of existing data prior to initialization. <<

**TEXT** Specifies the text that briefly describes the optical volume. More information on this parameter is in Commonly used parameters.

**\*BLANK:** Text is not specified. If the optical volume is being re-initialized, the text is not changed. *'description'*: Specify no more than 50 characters of text, enclosed in apostrophes.

**TYPE** Specifies the type of optical volume being initialized. Optical volumes for user applications are initialized as primary volumes. Backup optical volumes can be written to only by using the following set of optical backup commands: CVTOPTBKU, CPYOPT, and DUPOPT.

**\*PRIMARY:** The optical volume is used as a primary volume.

**\*BACKUP:** The optical volume is used as a backup volume.

**CCSID**

Specifies the character set in which the optical volume, directory, file names, and volume description are written. This parameter does not affect how user data is written. The user application must determine the character set in which the file data is written.

**\*CALC:** The system default character set is used. For the current release, this value is 500.

**500:** The EBCDIC character set and code page 500 are used.

**850:** The ASCII character set and code page 850 are used.

**MEDFMT**

Indicates the media format to use when writing to the optical media. There are two media formats, either \*HPOFS or \*UDF. For a complete comparison of the two media formats please refer to the

Optical Support  book.

**\*MEDTYPE:** Specifies that the operating system will determine which media format is used to initialize the volume. Which media format is used is based upon the optical media type.

- If the media type is \*WORM or \*UNKNOWN, the media will be initialized using the \*HPOFS format.
- If the media type is \*ERASE and has not been previously initialized the media will be initialized using the \*HPOFS format.
- If the media type is \*ERASE and has been previously initialized it will be initialized using the previous media format.
- If the media type is \*DVD-RAM, the media will be initialized using the \*UDF format.

**\*HPOFS:** The High Performance Optical File System (HPOFS) media format is used to initialize the volume. One of the characteristics of HPOFS is space occupied by a deleted file is not reused. The only way deleted file space can be recovered is to re-initialize the media >> thereby losing all previously recorded data on the media. <<

**\*UDF:** The Universal Disk Format (UDF) media format which is a subset of the ISO-13346 standard is used to initialize the volume. One of the characteristics of UDF is space occupied by a deleted file will be reused when needed for either the creation of a new file or the extension of an existing file. The UDF media format also provides file level security through permissions.

### **Example for INZOPT**

```
INZOPT VOL(VOL01) THRESHOLD(99) CHECK(*NO)
```

This command initializes the optical volume VOL01 with a volume-full-threshold of 99 percent. The system does not check to see if the volume is initialized.

### **Error messages for INZOPT**

#### **\*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.

##### **OPT1342**

Invalid volume identifier specified.

##### **OPT1345**

No free space available on media.

##### **OPT1346**

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

##### **OPT1350**

Write operations failed to optical volume &1.

##### **OPT1360**

Media directory corrupted on optical volume &1.

##### **OPT1375**

Optical volume &1 already exists.

##### **OPT1460**

Optical volume &1 is not in an optical device.

##### **OPT1485**

Initialize or rename of optical volume failed.

##### **OPT1489**

Volume parameter is not permitted for device &1.

- OPT1530**  
&1 does not represent a valid optical device.
- OPT1540**  
Invalid parameters specified.
- OPT1555**  
Optical device &1 in use.
- OPT1605**  
Media or device error occurred.
- OPT1790**  
Operation not allowed or 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.
- OPT2301**  
Internal system object in use.
- OPT2420**  
Not authorized to optical volume &2.
- OPT2422**  
Not authorized to file or directory.
- OPT7740**  
User not authorized to object &2 in library &3 type &4.

---

## **INZPFM (Initialize Physical File Member) Command Description**

INZPFM Command syntax diagram

### **Purpose**

The Initialize Physical File Member (INZPFM) command initializes records in a member of a physical file to the specified type of record (either default or deleted records). This command is usually used for files that are processed in arrival sequence or by relative record numbers. If the initialized member is empty, records are added and initialized to the specified type; if the member is not empty, records of the specified type are added to the member. As many records are added as is necessary to make the total record count specified.

**Note:** The INZPFM command ignores all file overrides that are currently in effect for the job.

### Restrictions:

1. The member may be open for input (read only) while the initialize operation takes place but cannot be open for update, delete, or insert.
2. To initialize the member with default records, the user needs object operational authority, object management authority or alter, and add authority for the file in which the member exists and execute authority to the library.
3. To initialize the member with deleted records, the user also needs delete authority for the file.
4. An \*EXCLRD lock is required on the member to initialize it.
5. In multithreaded jobs, this command is not threadsafe and fails for Distributed Data Management (DDM) files of type \*SNA.

### Required Parameter

**FILE** Specifies the qualified name of the physical file that contains the member being initialized.

The name of the 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.

*physical-file-name:* Specify the name of the physical file.

### Optional Parameters

**MBR** Specifies the name of the member that is initialized.

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

**\*LAST:** The last member of the specified physical file is initialized.

*physical-file-member-name:* Specify the name of the physical file member that is initialized.

### RECORDS

Specifies the type of records that are initialized (added) to the specified member. The specified members are initialized with default records or deleted records.

**\*DFT:** The specified member is initialized with default records. If a default value was specified in the DDS (DFT keyword) for a field, that field is initialized to the specified default; otherwise, all numeric fields are initialized to zeros and all character fields are initialized to blanks.

**\*DLT:** The specified member is initialized with deleted records. The records are not eligible for access, but simply hold a place in the file. Deleted records can be updated to reuse the deleted space.

## **TOTRCDS**

Specifies the total number of records in the member after it is initialized. If the value specified in this parameter causes the size of the file to be larger than the size specified when the file was created, a message is sent to the system operator's message queue (QSYSOPR). The operator can either continue or end the operation.

**\*NXTINCR:** The number of records in the member is increased to extend the file to the next allocation increment. If the member is empty, records are added to meet the first allocation specified for the member. If SIZE(\*NOMAX) is specified when the file is created, \*NXTINCR is not valid.

*total-records:* Specify the total number of records, ranging from 1 to 4294967288, by which the member is increased. If the number of existing records in the member already meets or is larger than this number, no records are initialized; if the number is less than that specified, enough records are initialized to equal the total specified.

## **Example for INZPFM**

```
INZPFM FILE(*CURLIB/INV) TOTRCDS(12000)
```

This command initializes as many as 12,000 records in the first member of the physical file named INV in the job's current library \*CURLIB. Only the number of records are added that brings the total to 12,000 records in the member. Any records that are added are initialized to the default format. If a default value is specified in the DDS (DFT keyword) for a field, that field is initialized to the specified default; otherwise, all numeric fields are initialized to zeros and all character fields are initialized to blanks.

## **Error messages for INZPFM**

### **\*ESCAPE Messages**

#### **CPF3130**

Member &2 already in use.

#### **CPF3131**

Cannot initialize member &2 with default records.

#### **CPF3132**

TOTRCDS parameter value either missing or too small.

#### **CPF3133**

File &1 in library &3 contains no members.

#### **CPF3134**

Referential constraint error processing member &2.

#### **CPF3136**

File &1 in &3 not allowed on command.

#### **CPF3137**

No authority to clear, initialize, or copy member &2.

#### **CPF3138**

Check constraint error processing member &2.

#### **CPF3140**

Initialize or copy of member &2 canceled.

#### **CPF3141**

Member &2 not found.



- CPF3142**  
File &1 in library &3 not found.
- CPF3143**  
Increments not allowed for member &2.
- CPF3144**  
Member &2 not cleared or initialized.
- CPF3148**  
New records need too much space for member &2.
- CPF3156**  
File &1 in library &3 in use.
- CPF3157**  
Triggers prevent requested operation.
- CPF3159**  
Member &2 saved with STG(\*FREE).
- CPF3160**  
Operation on member &2 ended. Entry cannot be journaled.
- CPF3179**  
Cannot clear or initialize DDM file &1 in &3.
- CPF3180**  
Member &2 not initialized.
- CPF32CF**  
Distributed file error, reason code &3.
- CPF32C3**  
Distributed file error, level ID mismatch
- CPF320B**  
Operation was not valid for database file &1.
- CPF9801**  
Object &2 in library &3 not found.
- CPF9810**  
Library &1 not found.
- CPF9820**  
Not authorized to use library &1.


---

## INZSYS (Initialize System) Command Description

INZSYS Command syntax diagram

### Purpose

The Initialize System (INZSYS) command initializes conversions done during installation procedures. This process is initiated during the first IPL after the software package is installed.

More information is available in the Software Installation  book.

### Optional Parameter

#### MSGQ

Specifies the qualified name of the message queue to which messages are sent.

**\*SYSOPR:** Messages from the system operator message queue (QSYSOPR) are sent to the system operator.

The name of the message queue 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.

*message-queue-name:* Specify the name of the message queue name to which messages are sent.

### Example for INZSYS

INZSYS

This command initializes the conversions done during installation procedures.

### Error messages for INZSYS

#### \*ESCAPE Messages

##### CPF372A

INZSYS or GO LICPGM currently running in another job.

##### CPF90E2

Error occurred for previous release file &1 in library &2.

##### CPF90E3

Error occurred for file &1 in library &2.

##### CPF90E4

System function in use. Reason code &1.

##### CPF90E8

Error occurred for file &1 in library &2.

##### CPF90E9

Data exists for more than one previous release.

---

## INZTAP (Initialize Tape) Command Description

INZTAP Command syntax diagram

### Purpose

The Initialize Tape (INZTAP) command prepares magnetic tapes for use on the system. This command is used to write volume labels on standard-labeled magnetic tapes so the tape device support can do standard-label processing. Unlabeled tapes must also be initialized by this command or by a similar process on another system before these tapes can be used on the iSeries 400. Note that this command, unlike the Initialize Diskette (INZDKT) command, does not indicate whether the tape is used for data written in the basic exchange format or in the save and restore format. That is done when the data files are actually written on the tape.

Three operations can be done by this command, depending on the values specified for the CHECK and CLEAR parameters.

1. The tape can be checked for active data files (files that have not reached their expiration dates).
2. The tape can be initialized either as a standard-labeled tape (if NEWVOL specifies a volume identifier) or as an unlabeled tape. Initialization writes only on the beginning of the tape, but it makes all data on the reel inaccessible.
3. If the tape is being initialized, any previous data on the tape can also be deleted.

If the tape is being initialized as a standard-labeled volume, a volume label followed by two tape markers is written at the beginning of the tape. If it is initialized as an unlabeled tape, only the two tape markers are written at the beginning of the tape.

All tapes must be initialized before use. Tapes that have been initialized do not need to be reinitialized unless the user wants to write a new volume label, change the tape type from a standard-labeled tape to an unlabeled tape or vice versa, or change the density of a standard labeled tape.

### Required Parameter

**DEV** Specifies the name of the device in which the volume being initialized is placed. Specify the name of the tape or media library device.

### Optional Parameters

#### NEWVOL

Specifies the volume identifier for a tape being initialized as a standard-labeled tape. If no volume identifier is specified, the tape is initialized as an unlabeled tape; that is, it has no volume label and no header labels for data files that are written on it later.

#### 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.

**\*NONE:** The tape is initialized as an unlabeled tape. Only tape markers are used to indicate the beginning and end of each data file on it, and the beginning and end of the volume itself.

**\*CTGID:** The tape is initialized as a standard labeled tape. The new logical volume identifier is the same as the external identifier of the tape cartridge. Each tape within a library device must have a unique external identifier.

*new-volume-identifier:* Specify up to 6 characters to identify the new volume. The identifier must contain only alphanumeric characters (A through Z, \$, #, @, and 0 through 9), and cannot have a prefix or contain blanks. Each tape reel should have a unique volume identifier to ensure optimum protection and control of the tape volumes.

#### NEWOWNID

Specifies which tape owner's identifier to write in the volume label of the volume being written. The owner identification contains up to 14 characters (letters and/or numbers in any combination), is left-justified, and padded on the right with blanks if fewer than 14 characters are supplied.

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

*owner-identifier:* Specify up to 14 characters that identify the owner of the tape. If fewer than 14 characters are specified, the field is left-justified and padded on the right with blanks.

**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:** Any labeled or unlabeled volume on the specified tape device is initialized. VOL(\*MOUNTED) and CHECK(\*NO) must be used to initialize a new or completely erased tape volume. Otherwise, the system attempts to read labels from the tape volume that is on the specified device and completely unwinds from the reel. 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 initialized. This parameter value can be used only to reinitialize a tape that is already a labeled volume. If the tape on the specified device has a different volume identifier than the one specified by this value, or if it is an unlabeled volume, an error message is sent.

**CHECK**

Specifies whether a labeled tape volume is checked for active data files (files with an end date later than the current system date) before it is initialized. If an unlabeled volume is on the specified device for initialization, this parameter is ignored. If the volume must be checked for active files, as much of the tape is read as necessary before initialization is done.

**\*YES:** All data file labels on the tape are checked. If active files are found, the operation is ended and an error message is sent.

**\*NO:** Tape initialization continues with no checking for active files. To initialize a new or empty volume, VOL(\*MOUNTED) and CHECK(\*NO) must be specified; otherwise, the system attempts to read labels from the volume on the specified device until the tape completely unwinds from the reel.

**\*FIRST:** Only the first data file label on the tape is checked. If there are no data files in the volume, or if the first data file has ended, the volume is initialized without checking for any other files on the reel. If the first data file has not ended (it is active), the operation is stopped and an error message is sent.

If it is known that there is only one data file on the tape, that there are no active files on the volume, or that all files have the same end date, use CHECK(\*FIRST) to do the fastest initialization of the tape. Note, however, that any data files past the first one are destroyed, with no end check made.

**DENSITY**

Specifies the density or format in which to write the data on the tape after it has been initialized. The density used for all data files written to a standard-labeled tape is specified when the volume is initialized, and cannot be changed unless the tape is reinitialized. For a tape that is not labeled, the DENSITY parameter specified in the Create Tape File (CRTTAPF), the Change Tape File (CHGTAPF), or the Override Tape File (OVRTAPF) command can change the density of the volume when the first data file on the tape is created.

**\*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.

**\*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

*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.

**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 extended binary-coded decimal interchange code (EBCDIC) character set code is used.

**\*ASCII:** The ASCII character set code is used.

#### **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.

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

#### **CLEAR**

Specifies whether all previous labels and data are deleted from the tape when it is initialized. If the volume must be cleared of all data, it is spaced from the location of the initializing volume label or tape markers to the end of the tape marker.

**\*NO:** The existing data is not deleted. Even though the existing data is not deleted, the data on the volume is not accessible after the volume has been initialized.

**\*YES:** After the beginning of the tape has been initialized, the remaining data on the tape is deleted. The \*YES value is needed only if there are security concerns with old data. If \*YES is specified, the initialize operation can take a long time.

#### **Example for INZTAP**

```
INZTAP DEV(TAPE1) NEWVOL(T00100) CHECK(*NO)
      CODE(*ASCII) ENDOPT(*UNLOAD)
```

This command initializes the volume on the tape device named TAPE1 using the ASCII character code. Its new volume identifier is T00100, regardless of whether it contains a valid volume identifier or files that have not ended (active field). Once the volume has been initialized, the tape is rewound and unloaded. Any previous data beyond the new volume label is not deleted, but is no longer accessible.

#### **Error messages for INZTAP**

##### **\*ESCAPE Messages**

###### **CPF67A0**

Volume ID does not match cartridge ID

###### **CPF6702**

Error processing volume on device &1.

###### **CPF6708**

Command ended due to error.

###### **CPF6715**

Error at beginning of tape on device &1.

###### **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.



**CPF6745**

Device &1 not a media library device.

**CPF6750**

NEWVOL(\*NONE) not valid for device &1.

**CPF6751**

Load failure occurred on device &4.

**CPF6754**

Active file &4 found on volume &2.

**CPF6760**

Device &1 not ready.

**CPF6762**

Wrong type of cartridge in device &1.

**CPF6763**

Wrong type of cartridge in device &1.

**CPF6768**

Volume on device &1 is write protected.

**CPF6772**

Volume on device &1 cannot be processed.

**CPF6774**

New volume &2 is a nonstandard labeled tape. Volume not prepared.

**CPF9814**

Device &1 not found.

**CPF9825**

Not authorized to device &1.

---

## INSPTF (Install Program Temporary Fix) Command Description

INSPTF Command syntax diagram

### Purpose

The Install Program Temporary Fix (INSPTF) command allows the user to load and apply PTFs for multiple products with a single command. >> PTF groups will be copied to the system when they do not already exist on the system or when the level of the PTF group on the media is higher than the level of the PTF group that exists on the system. <<

The OMIT and HIPER parameters are supplied to allow the user of the INSPTF command to be more selective. These parameters apply only to the PTF loading activity. Any PTF already loaded on the system will be applied.

The INSTYP parameter controls the apply of the PTFs. Using the different special values allows immediate and delayed apply combinations as well as starting an IPL.

INSPTF does not support loading PTFs for products that have multiple releases installed on the system. If PTFs for such a product exist on the media, the INSPTF will not load those PTFs and will return an error.

### Required Parameter

## LICPGM

Specifies the product ID, version, release, and modification level of the products for which PTFs should be installed.

**\*ALL:** The available PTFs for all installed products are installed. This must be the first and only value if specified. All values specified after it are ignored.

### Element 1: Licensed Program

*licensed-program:* Specify the product ID of the PTFs to be installed.

### Element 2: Release Level of the Licensed Program

*release-level:* Specify the release level of the base product option in the format VxRyMz, where Vx is the version number, Ry is the release number, and Mz is the modification level.

**\*ONLY:** This value is valid only when one release of the product's base option is installed on the system. PTFs for all installed options of the product are loaded and applied regardless of the release-level of the option.

## Optional Parameters

**DEV** Specifies the device from which the PTFs are loaded. The device name must already be known on the system by a device description.

**\*SERVICE:** The PTFs sent from the service support system are loaded.

*tape-device-name:* Specify the name of the tape device from which the PTFs are loaded.

*optical-device-name:* Specify the name of the optical device from which the PTFs are installed.

## INSTYP

Specifies the type of install to perform.

**\*SRVATT:** The type of install depends on the service attribute setting.

<b>CAUTION:</b>
The service attribute is shipped with *DLYIPL as the default. Use the Change Service Attributes (CHGSRVA) command to change the default.

**\*DLYIPL:** All PTFs are marked for delayed apply and an initial program load (IPL) is done on the system.

**\*DLYALL:** All PTFs are marked for delayed apply and an initial program load (IPL) is not done on the system.

**\*IMMONLY:** Only the immediate PTFs are applied and an initial program load (IPL) is not done on the system.

**\*IMMDLY:** The immediate PTFs are applied and the delayed PTFs are marked for apply at the next initial program load (IPL).

## HIPER

Specifies whether only high-impact pervasive (HIPER) PTFs should be loaded when installing from a media.

### Note:

This parameter is ignored if DEV(\*SERVICE) is specified.

This is only valid when installing IBM cumulative PTF packages.

**\*NO:** All PTFs, other than those listed in the omit list, should be installed.

**\*YES:** Only HIPER PTFs that are not on the omit list should be loaded.

**OMIT** Specifies the product ID, version, release, modification level, and PTF ID for PTFs that should not be loaded. The current state of the PTF is not checked before being passed to LODPTF. If the PTF is already loaded it is applied.

**Element 1: Licensed-Program**

*licensed-program:* Specify the product ID for the PTFs that should not be loaded.

**Element 2: PTF-Number**

*PTF-number:* Specify the PTF ID for the PTFs that should not be installed.

**Element 3: Release Level**

*release-level:* Specify the release level of the base product option or the release level of the PTF for the PTFs that should not be loaded. The release level must be specified in VxRyMz format, where Vx is the version number, Ry is the release number, and Mz is the modification level.

**\*ONLY:** The only release of the licensed-program selected on the LICPGM parameter.

**ENDOPT**

Specifies the operation that is automatically performed on the tape or optical volume after the PTF operation ends.

**Note:**

This parameter is valid only if a tape or optical device name is specified on the DEV parameter. For optical devices, \*UNLOAD is the only special value supported, \*REWIND and \*LEAVE will be ignored.

**\*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. Some optical devices will eject the volume after the operation ends.

**IPLTYPE**

Specifies the point from which the initial program load (IPL) restarts when the PTF apply type (INSTYP) parameter indicates an IPL will be performed.

**Note:**

This is valid only when INSTYP(\*DLYIPL) is specified or when INSTYP(\*SRVATT) is specified and the PTF install type (PTFINSTYP) service attribute is set to \*DLYIPL.

**\*IPLA:** The value specified on the Change IPL Attributes (CHGIPLA) command is used. To determine the current setting for this value, use the Display IPL Attributes (DSPIPLA) command.

**\*SYS:** Specifies that the system determines how much of the system to restart.

The operating system is always restarted. The hardware is restarted only if a PTF that requires a restart is applied. Other hardware functions, such as some configuration changes, that occur during a \*FULL IPL are not processed.

\*SYS can result in a shorter IPL time than if you specify \*FULL.

\*FULL: All portions of the system, including the hardware, are restarted.

## Examples for INSPTF

### Example 1: Omitting PTFs

```
INSPTF LICPGM((*ALL)) DEV(*SERVICE) INSTYP(*IMMDLY)
      OMIT((5722999 MF12345 V5R2M0) (5722SS1 SI12345 V5R2M0))
```

This command will load all PTFs that are in \*SERVICE for all products installed on the system except MF12345 and SI12345. It will then apply all PTFs in loaded status that can be applied immediately and mark the rest for delayed apply.

### Example 2: Installing HIPER only

```
INSPTF LICPGM((5722PT1 V5R2M0)) DEV(TAP01)
      INSTYP(*IMONLY) HIPER(*YES)
```

This command will search the media for PTFs for the V5R2M0 Performance Tools product in the HIPER section. Each PTF that can be applied immediately will be. Delayed PTFs will be loaded, but not marked for apply.

### Example 3: Installing Only Immediate PTFs

```
INSPTF LICPGM((*ALL)) DEV(TAP01) INSTYP(*IMONLY)
      ENDOPT(*LEAVE)
```

This command will load all PTFs for the products that are installed on the system, from the device TAP01. All PTFs in loaded status on the system that can be applied immediately will be. No delayed PTFs will be set for apply.

## Error messages for INSPTF

### \*ESCAPE Messages

#### CPF3586

List of PTFs not correct.

#### CPF358A

Release not valid.

#### CPF358F

LICPGM parameter contains duplicate entries.

#### CPF35EB

Multiple releases of product &1 installed.

#### CPF3615

PTF install processing failed.

#### CPF3618

The mode is not set at Normal.

#### CPF361A

PTFs installed successfully, but actions pending.

#### CPF361B

PTF install processing failed, and there are actions pending.

#### CPF361C

No PTFs installed.



---

## INSRMPRD (Install Remote Product) Command Description

**Note:** To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

INSRMPRD Command syntax diagram

### Purpose

The Install Remote Product (INSRMPRD) command provides the capability to install a product, packaged using the System Manager licensed program, from the central site system on one or more managed systems.

**Note:** A change request is automatically submitted which can be viewed to determine the status of this command. A message is returned identifying the name of the change request.

### Restrictions:

1. This command is shipped with public \*EXCLUDE authority.
2. To run this command, you must have the authority necessary to use the Add Product Change Request Activity (ADDPRDCRQA) command.
3. If a NODL value is specified, the node list can only contain entries that have a value of \*SNA for the address type.
4. The Managed System Services licensed product and the OS/400 licensed product cannot be specified.

### Required Parameter

#### PRDID

Specifies the 7-character identifier of the product to be installed.

*product-ID:* Specify the 7-character product ID that is installed.

### Optional Parameters

**RLS** Specifies which version, release, and modification level of the product is used.

**\*ONLY:** The release level is determined by searching the system for a product definition for the given product ID. The release level is taken from the product definition. This value is not valid if more than one product definition exists for the same product ID.

*version-release-modification:* Specify the release level in the format VxRxMy, where Vx is the version number, where Rx is the release number, and My is the modification number. Valid values for x are the numbers 0 through 9. Valid values for y are the numbers 0 through 9 and the letters A through Z.

#### OPTION

Specifies which of the optional parts of the product given in the PRDID parameter are used.

**\*BASE:** Only the base part of the product is used.

*product-option-number:* Specify the option number for the product load being saved. Valid values range from 1 through 99.

#### LODTYPE

Specifies the product load objects being used.

**\*ALL:** Code and language objects specified on the LODID parameter are used.

**\*CODE:** The object associated with this product load are used.

**\*LNG:** The objects associated with the national language version (NLV) identified on the LODID parameter are used.

## LODID

Specifies the load identifier used.

**\*ALL:** All languages for this product option are used.

**\*CODE:** The code load is used.

*product-load-ID:* Specify when LODTYPE(\*LNG) or LODTYPE(\*ALL) is specified, the load ID must be one of the valid IBM national language versions and be specified in the form 29xx, where x is any value from 0 through 9.

**NODL** Specifies that the node list parameter is the object name that contains a list of systems that are the destinations for the activity. It cannot be specified if the control point name (CPNAME) parameter is also specified.

**\*NONE:** The systems on which this activity is to be performed are not specified by a node list. Individual control point names must be specified.

The possible library values are:

**\*LIBL:** All of the libraries in the user and system portions of the job's library list are searched for the node list object.

**\*CURLIB:** The current library for the job is used to locate the node list object.

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

*node-list-name:* Specify the node list object name containing the list of systems on which the activity is to be performed.

## CPNAME

Specifies the APPN control point names of the managed systems on which this activity is to be performed. Control point names cannot be specified if the node list (NODL) parameter is specified. If ACTION(\*RTV) is used, only one control point name can be specified.

**\*NONE:** The systems on which this request is performed are not identified individually. A node list must be specified.

The possible network identifier values are:

**\*NETATR:** The network ID of the local system is used. This is useful when the node being specified is in the same network as the local system.

*network-identifier:* Specify the APPN network identifier of the managed system on which the request is to be performed.

*control-point-name:* Specify the APPN control point name of the managed system on which the request is to be performed.

## TGTRLS

Specifies the release of the operating system on which you intend to use the object. This parameter is ignored for objects with global names that are in the SystemView distribution repository or for actions other than send or retrieve.

**\*CURRENT:** The object is used on the release of the operating system currently running on your system. If V5R2M0 is running on your system, \*CURRENT means that you intend to use the object on a system with V5R2M0 installed. The object can also be used on a system with any later release of the operating system installed.

**\*PRV:** The object is intended for a system that is at the previous release level compared to the local system.

### Note:

Modification levels are not supported. To specify the previous release with a modification level other than 0, such as V4R1M4, specify the value for the release rather than the special value \*PRV.

*release-level:* Specify the release level in the VxRxMx format. The object is used on a system with the specified release or with any later release of the operating system installed.

Valid values depend on the current version, release, and modification level and they change with each new release.

## KEEPCLGE

Specifies if the distribution catalog entry and its associated save file corresponding to the product is kept on the specified system.

**\*NO:** The catalog entry and associated save file are not kept.

**\*YES:** The catalog entry and associated save file are kept.

## Examples for INSRMTPRD

### Example 1: Installing a Product

```
INSRMTPRD PRDID(1ACCOUN) RLS(V5R2M0) OPTION(*BASE)
LODTYPE(*ALL) LODID(*ALL) CPNAME((*NETATR SYS1))
```

This command installs 1ACCOUN product with the release V5R2M0 of the base option for both the program and the language parts on the iSeries SYS1.

### Example 2: Installing an Option

```
INSRMTPRD PRDID(1CHECKS) RLS(V5R2M0) OPTION(10)
LODTYPE(*ALL) LODID(*ALL) CPNAME((*NETATR SYS2))
```

This command installs option 10 of the 1CHECKS product, release V5R2M0, for both the program and the language parts on the iSeries SYS2.

## Error messages for INSRMTPRD

### \*ESCAPE Messages

None <<

---

## LNKDTADFN (Link Data Definition) Command Description

LNKDTADFN Command syntax diagram

### Purpose

The Link Data Definition (LNKDTADFN) command links or unlinks file definitions in a dictionary, program-described files, or externally-described files.

**Restriction:** A file cannot be linked if it is already linked. However, a definition can be linked to several files at the same time.

**Note:** If file text and the file definition are not the same, a new version of the definition is created.

### Required Parameters

#### OPTION

Specifies the action performed on the program-described file, externally-described file, or file definition.

**\*LINK:** The program-described file or the file definition is linked.

**\*UNLINK:** The program-described file, the externally-described file, or the file definition is unlinked.

**FILE** Specifies the qualified name of the program-described file to link or unlink, or specifies the externally-described file to unlink.

**\*ALL:** All program-described database files linked to definitions in the specified dictionary are unlinked. This value is valid only if OPTION(\*UNLINK) is also specified. This value is not valid for externally-described files.

The name of the program-described 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 file being linked or unlinked.

### Optional Parameters

#### DTADCT

Specifies the name of the dictionary that contains the file definition to link to the program-described file, or the dictionary from which all program-described files are unlinked.

This parameter is required for the \*LINK option. It is also required if FILE(\*ALL) is specified and the option is \*UNLINK.

**DFN** Specifies the name of the file definition that is linked to the program-described file. This parameter is only required with the \*LINK option.



## **CRTDATE**

Specifies the creation date of the object.

**Note:** This parameter is valid only with the \*LINK option.

**\*FIRST:** The file definition with the specified definition name and the earliest creation date is used.

*date:* Specify the creation date of the file definition to link to the program-described file. If a date is not specified, \*FIRST is assumed.

### **Example for LNKDTADFN**

```
LNKDTADFN OPTION(*LINK) FILE(MYLIB/MYFILE)
          DTADCT(MINE) DFN(MYDEF)
```

This command links definition MYDEF in dictionary MINE, to the program-described database file MYFILE located in library MYLIB.

### **Error messages for LNKDTADFN**

#### **\*ESCAPE Messages**

##### **CPF2E9B**

Definition &1 not found.

##### **CPF2FE0**

Error occurred while opening dictionary &1.

##### **CPF2FE1**

Error occurred while closing dictionary &1.

##### **CPF2FE2**

Dictionary &1 currently in use.

##### **CPF2FE3**

System cross reference file is in error.

##### **CPF2FE4**

System cross reference file not available.

##### **CPF2F02**

Not authorized to use dictionary &1.

##### **CPF2F07**

Dictionary &1 in error.

##### **CPF2F08**

Dictionary &1 not found.

##### **CPF2F6A**

File &2 in &3 not valid for LNKDTADFN.

##### **CPF2F6C**

All files were not unlinked.

##### **CPF2F61**

File &2 in &3 currently in use.

##### **CPF2F7B**

File &2 not linked. Record lengths not equal.

**CPF2F7C**

Start key position &1 splits field &2.

**CPF2F7D**

End key position &1 splits field &2.

**CPF2F7F**

File &2 in &3 is already linked.

**CPF2F76**

Only file definitions for physical files can be linked.

**CPF2F77**

File not keyed. Cannot link to keyed file definition.

**CPF2F78**

Definition &1 in error.

**CPF2F79**

Key fields do not match.

**CPF2F80**

File &2 in &3 is not linked.

**CPF9812**

File &1 in library &2 not found.

**CPF9820**

Not authorized to use library &1.

**CPF9822**

Not authorized to file &1 in library &2.

**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.

---

## LODRUN (Load and Run Media Program) Command Description

LODRUN Command syntax diagram

### Purpose

The Load and Run Media Program (LODRUN) command restores a user-written program object from tape, diskette, or optical device into the library QTEMP. The system passes the device name to the restored program and transfers control to the restored program.

**Restriction:** The LODRUN command requires \*SAVSYS authority. The QINSTAPP program that is loaded from the media and called may require additional authority in order to run correctly. The user supplying the QINSTAPP program should inform you if any additional authorities are required.

When the LODRUN command is run:

1. The media is searched for the user-written program, which must be named QINSTAPP and saved from library QTEMP.

**Note:**

The program QINSTAPP must be owned by a user profile that resides on the target system. If QINSTAPP is restored to a system that does not have the owning user profile, control is not transferred and the program is not run.

2. If a QINSTAPP program already exists in the QTEMP library on the user's system, it is deleted.
3. The QINSTAPP program is restored to the QTEMP library using the RSTOBJ command. The following values are specified on the RSTOBJ command:
  - OBJ(QINSTAPP)
  - OBJTYPE(\*PGM)
  - SAVLIB(QTEMP)
  - ENDOPT(\*LEAVE)
  - MBROPT(\*ALL)
  - ALWOBJDIF(\*NONE)
  - RSTLIB(QTEMP)
  - VOL(\*MOUNTED)

If the device is an optical device, the ENDOPT and the VOL parameters are not specified.

If the device is optical, then the value specified for the DIR for this command is used for the OPTFILE parameter for the Restore Object (RSTOBJ) command.

The SEQNBR parameter is specified according to the SEQNBR parameter on the LODRUN command.

The device used for the restore operation is determined by the LODRUN command. If \*TAP, \*DKT, or \*OPT are specified on the DEV parameter, the LODRUN command examines the QDEVNAMING system value to determine if the system uses iSeries 400 or System/36 naming conventions for devices:

- If QDEVNAMING is \*NORMAL (iSeries 400 convention)
  - The device TAP01 is used for DEV(\*TAP).
  - The device DKT01 is used for DEV(\*DKT).
  - The device OPT01 is used for DEV(\*OPT).
- If QDEVNAMING is \*S36 (System/36 convention)
  - The device TC is used for DEV(\*TAP) if a tape cartridge is found. Otherwise, device T1 is used.
  - The device I1 is used for DEV(\*DKT).
  - The device OPT01 is used for DEV(\*OPT). System/36 naming conventions do not apply to optical devices.

Any other value specified on the DEV parameter is used as is.

4. Control of the system is passed to the QINSTAPP program. The QINSTAPP program can be used, for example, to restore other applications to the user's system and run those applications.
5. When the user signs off, the QINSTAPP program is removed from the system.

**Note:**

The settings of three system values work together to determine if the QINSTAPP program is allowed to be restored or if it is converted during the restore. The three systems values are:

- QVfyOBJRST Verify object on restore
- QFRCCVNRST Force conversion on restore

- QALWOBJRST Allow object restore option



The user supplying the QINSTAPP program is responsible for writing and supporting it. The QINSTAPP program is not supplied by IBM. The program can be designed to accomplish many different tasks. For example, the program could:

- Restore and run other programs or applications
- Restore a library
- Delete another program or application
- Create specific environments
- Apply fixes to existing applications

The QINSTAPP program is run only once each time the LODRUN command is entered. The LODRUN command passes only one parameter (DEV), which specifies the device from which the QINSTAPP program is restored. The QINSTAPP program should not attempt to use the LODRUN command again. This will have unpredictable results.

In addition to writing the QINSTAPP program, the user supplying the program is responsible for providing the user with the media containing the program. To distribute the program on a tape, diskette, or optical device, do the following:

1. Prepare the tape or diskette. For tape, use the Initialize Tape (INZTAP) command. For diskette, use the Initialize Diskette (INZDKT) command with FMT(\*SAVRST) specified.
2. Use the Create Duplicate Object (CRTDUPOBJ) command to create the QINSTAPP program into the QTEMP library.
3. Use the Save Object (SAVOBJ) command to save the QINSTAPP program from QTEMP to the desired tape device or diskette unit. The program must be the only object in the media file that contains it. Specify the following:

```
LIB(QTEMP)
```

```
LABEL(*LIB)
```

```
CLEAR(*ALL)
```

Specifying LABEL(\*LIB) ensures that the label will be QTEMP. If the QINSTAPP program is being saved to a tape device, and if additional applications, programs, or libraries will be saved to tape, ENDOPT(\*LEAVE) also must be specified. The correct value for the TGTRLS parameter must also be entered if the target release is not the default, \*CURRENT.

4. Use the Save Object (SAVOBJ), Save Library (SAVLIB), or Save License Program (SAVLICPGM) command to save any other necessary applications, programs, or libraries to the tape or diskette. This step is optional and is used to save applications that the QINSTAPP program restores to the user's system when the LODRUN command is run.

If the QINSTAPP program is saved to tape, the tape is not rewound after the QINSTAPP program is restored; the application or series of applications that the QINSTAPP program restores to the user's system should be next on the tape.

### Optional Parameters

**DEV** Specifies the I/O device from which the program is loaded.

**\*TAP:** The program is loaded from the default tape device connected to the system.

**\*DKT:** The program is loaded from the default diskette device connected to the system.

**\*OPT:** The program is loaded from the default optical device connected to the system.

*tape-device:* Specify the name of the tape device from which the program is loaded onto the system.

*diskette-device:* Specify the name of the diskette unit from which the program is loaded onto the system.

*optical-device:* Specify the name of the optical device from which the program is loaded onto the system.

## **SEQNBR**

Specifies, only when tape is used, the sequence number used for the restore operation.

**\*FIRST:** The volume on a tape device is searched starting from the first data file for a data file with an identifier that is a match for the QTEMP label. When the first match is found, the object is restored.

**\*SEARCH:** The volume on a tape device is searched starting from the first data file beyond the current tape position for a data file with an identifier that is a match for the QTEMP label. When a match is found, the object is restored.

*sequence-number:* Specify the sequence number of the file. Valid values range from 1 through 16777215.

**DIR** Specifies, only when an optical device is used, the directory used for the restore operation. If the file named QTEMP is found in the specified directory, the object is restored.

*/:* The root directory (/) is used.

*directory-name:* Specify the directory to search for a file named QTEMP.

## **Examples for LODRUN**

### **Example 1: Restoring a Program from Tape**

```
LODRUN  DEV(TAP01)
```

This command restores the program object from the tape on device TAP01 to the library QTEMP. Control is then transferred to the restored program.

### **Example 2: Restoring the Program QINSTAPP from Tape**

```
LODRUN  DEV(TAP01) SEQNBR(5)
```

This command restores the program object QINSTAPP from the tape at sequence number 5 on device TAP01 to the library QTEMP. Control is then transferred to the restored program. If the sequence number is not found, an escape message is sent. If the file label at that sequence number is not QTEMP, an escape message is sent.

### **Example 3: Restoring the Program QINSTAPP from CD-ROM**

```
LODRUN  DEV(*OPT) DIR(/APP1/INST)
```

This command restores the program object QINSTAPP from the CD-ROM on device OPT01 to the library QTEMP. The filename for the QTEMP library on the CD-ROM is /APP1/INST/QTEMP. Control is then transferred to the restored program. If the file is not found, an escape message is sent. >>

---

# LODIMGCLG (Load or Unload Image Catalog) Command Description

LODIMGCLG Command syntax diagram

## Purpose

The Load or Unload Image Catalog (LODIMGCLG) command is used to associate an image catalog and its images to a virtual optical device. The status of the image catalog will be changed based on the OPTION parameter as follows:

### OPTION(\*LOAD)

This will cause the status of the image catalog to change to Ready.

### OPTION(\*UNLOAD)

This will cause the status of the image catalog to change to Not ready.

Only one image catalog can be associated with a virtual optical device. If the virtual optical device already has an image catalog associated with it, you can use OPTION(\*UNLOAD) to unload the current image catalog.

## Restriction:

1. You must have \*SECADM and \*ALLOBJ special authorities to use this command.

## Required Parameters

### IMGCLG

Specifies the name of the image catalog to be loaded into or unloaded from the virtual optical device.

*image-catalog*: Specify the image catalog to be loaded into or unloaded from the virtual optical device.

**DEV** Specifies the name of the virtual optical device that the image catalog is to be loaded into or unloaded from.

*device-name*: Specify the name of the device that the image catalog is to be loaded into or unloaded from.

### OPTION

Specifies whether the image catalog is to be loaded or unloaded.

**\*LOAD**: The image catalog will be loaded into the virtual optical device.

**\*UNLOAD**: The image catalog will be unloaded from the virtual optical device.

## Examples for LODIMGCLG

### Example 1: Loading an Image Catalog

```
LODIMGCLG IMGCLG(MYCLG) DEV(OPTVRT01)
```

This command loads image catalog MYCLG into device OPTVRT01.

### Example 2: Unloading an Image Catalog

```
LODIMGCLG IMGCLG(MYCLG) DEV(OPTVRT01) OPTION(*UNLOAD)
```

This command unloads image catalog MYCLG from device OPTVRT01.

## Error messages for LODIMGCLG

### \*ESCAPE Messages

**CPFBC10**

Image catalog &1 not loaded.

**CPFBC11**

Image catalog &1 not unloaded.

**CPFBC40**

Not authorized to command &1.

**CPFBC41**

&1 command failed.



---

## LODPTF (Load Program Temporary Fix) Command Description

LODPTF Command syntax diagram

### Purpose

The Load Program Temporary Fix (LODPTF) command loads program temporary fixes (PTFs) for a specified product from a diskette, tape, optical volume, or save file into the product PTF library. Each PTF contains one or more objects, including programs, that can be applied to a product by the Apply Program Temporary Fix (APYPTF) command.

Only the PTFs for a single product, such as Operating System/400, can be loaded at a time. Not all of the PTFs for the specified program must be loaded, because specific PTFs can be selected or omitted and loaded.

**Restriction:** This command is shipped with public \*EXCLUDE authority and the QSRV user profile has private authority to use the command.

### Required Parameter

**LICPGM**

Specifies the 7-character identifier of the product for which the PTFs are loaded.

### Optional Parameters

**DEV** Specifies the device from which the PTFs are loaded. The device name must already be known on the system by a device description.

**\*SERVICE:** The PTFs sent from the service support system are loaded.

**\*SAVF:** The PTFs are loaded from a save file. If \*SAVF is specified, a save file name must be specified on the SAVF parameter.

*diskette-device-name:* Specify the name of the diskette device from which the PTFs are loaded.

*tape-device-name:* Specify the name of the tape device from which the PTFs are loaded.

*optical-device-name:* Specify the name of the optical device from which the PTFs are loaded.

**SEQNBR**

Specifies the sequence number on the tape volume where the load operation begins to load the PTF data.

**Note:**

This parameter is valid only if a tape device name is specified on the DEV parameter.

**\*SEARCH:** The tape volume is searched for the first program temporary fix (PTF) file for the specified licensed program.

*sequence-number:* Specify the sequence number of the PTF file being loaded. This sequence number must exist on the tape. Valid values range from 1 to 16777215.

## ENDOPT

Specifies the operation that is automatically performed on the tape or optical volume after the PTF operation ends.

### Note:

This parameter is valid only if a tape or optical device name is specified on the DEV parameter. For optical devices, \*UNLOAD is the only special value supported, \*REWIND and \*LEAVE will be ignored.

**\*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 operations ends. Some optical devices will eject the volume after the operation ends.

## PATHID

Specifies the number that identifies a file on the optical volume that contains the PTFs to be loaded. The PTF files for each product and release that exist on the optical media have a path identifier number to allow the files to be processed in a specific order. Only the PTFs from the specified path identifier are loaded on your system.

### Note:

This parameter is valid only if an optical device name is specified on the DEV parameter.

**\*FIRST:** The optical volume is searched for the first PTF file for the specified product and release, according to the search dependency specified on the SELECT parameter.

- When a specific PTF identifier is specified on the SELECT parameter, the first occurrence of the specified PTF is loaded.
- When \*ALL is specified on the SELECT parameter, the existing PTF file with the lowest path identifier is loaded.

**\*SELECT:** A list of the PTF files that exist on the optical volume that match the product and release is shown. You can select the specific file from which PTFs are loaded. This value cannot be selected in a batch environment.

*path-identifier:* Specify the path identifier of the existing PTF file from which to load the PTF data.

**SAVF** Specifies the qualified name of the save file from which the PTFs are loaded.

The name of the save 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.

*save-file-name:* Specify the save file name from which the PTFs are loaded.

## **SELECT**

Specifies which of the PTFs for the specified product are loaded. The OMIT parameter cannot be specified if SELECT is specified with PTF numbers.

### **Note:**

Permanently removed PTFs will be ignored when SELECT(\*ALL) and DEV(\*SERVICE) are specified. To load removed PTFs, specify the PTF number on the SELECT parameter.

**\*ALL:** All the PTFs for the specified product are loaded.

*PTF-number:* Specify the PTF identification numbers (up to 50) of the single PTFs being loaded.

**OMIT** Specifies that all PTFs, except those specified in this parameter, are loaded. Specify the PTF numbers of the PTFs that are omitted when the rest are loaded. Up to 50 PTF numbers can be specified. The OMIT parameter cannot be specified if single PTF numbers are specified on the SELECT parameter.

## **SPRPTF**

Specifies which operation is performed for temporarily applied PTFs that are superseded by PTFs encountered by this load operation.

**\*APYPERM:** For the specified product, any PTFs that are temporarily applied and are superseded by PTFs contained on the PTF media, are automatically permanently applied before loading the superseding PTFs. If the superseded PTFs have any prerequisite PTFs, they are also permanently applied by this operation.

**\*NOAPY:** The load operation stops when temporarily applied PTFs are being superseded by PTFs contained on the PTF medium. The temporarily applied PTFs that are superseded must be permanently applied (APYPTF command) or removed (RMVPTF command) before the LODPTF command can be processed again.

## **COVER**

Specifies whether to copy the cover letter for the PTF into a physical file.

### **Note:**

This parameter is valid only when a tape device name or optical device name is specified on the DEV parameter.

**\*YES:** After the PTF is loaded, the cover letter is copied into a physical file.

**\*NO:** The cover letter is not copied into a physical file.

**\*ONLY:** The cover letter is copied into a physical file but the PTF is not loaded. If the SEQNBR parameter is specified, it must contain the sequence number of the file that contains the PTF.

**RLS** Specifies the release level of the PTFs being loaded.

**\*ONLY:** This value is only valid when *one* release of the product's base option is installed on the system. PTFs for all installed options of the product will be loaded regardless of the release-level of the option.

*release-level:* Specify the release level in VxRyMz format, where Vx is the version number, Ry is the release number, and Mz is the modification level. The variables x and y can be a number from 0 through 9, and the variable z can be a number from 0 through 9 or a letter from A through Z.

If the release-level specified is the release-level of the base option of the product, PTFs for all installed options of the product are loaded regardless of the release-level of the option.

If the release-level specified is not the release-level of the base option of the product, only PTFs for the options installed at that release-level are loaded.

## Example for LODPTF

### Example 1: Omitting PTFs

```
LODPTF LICPGM(5722SS1)
      OMIT(SI00003 SI00008 SI00014)
```

This command loads all of the PTFs from the service support system (\*SERVICE) for the product 5722SS1 except SI00003, SI00008, and SI00014. The Apply Program Temporary Fix (APYPTF) command can then be used to apply these PTFs to the 5722SS1 product.

### Example 2: Selecting PTFs

```
LODPTF LICPGM(5722SS1) DEV(OPT01)
      SELECT(SI00009 SI00010)
```

This command loads the PTFs named SI00009 and SI00010 from the optical device named OPT01. The Apply Program Temporary Fix (APYPTF) command can then be used to apply these PTFs to the 5722SS1 product.

## Error messages for LODPTF

### \*ESCAPE Messages

#### CPF35AA

Licensed internal code PTF &2 already applied.

#### CPF35AB

Licensed Internal Code fix &2 not applied.

#### CPF35AE

Duplicate PTF &1 found.

#### CPF35A0

Cannot allocate library &1.

#### CPF35A1

Wrong copy of Licensed Internal Code in use.

#### CPF35A2

Required hardware changes not installed for PTF &2.

#### CPF35A3

Licensed Internal Code fix &2 not temporarily applied.

#### CPF35A5

Licensed Internal Code fix &2 not permanently applied.

#### CPF35A6

Language option &1 not installed for licensed program.

**CPF35A8**  
No PTFs to be loaded.

**CPF35A9**  
Error occurred while processing Licensed Internal Code fix.

**CPF35CF**  
PTF &1-&2 not applied.

**CPF35C1**  
LODPTF ended. No more storage available.

**CPF35C9**  
PTF &1-&2 &3 not valid.

**CPF35EB**  
Multiple releases of product &1 installed.

**CPF35E3**  
Interface error detected.

**CPF35FA**  
PTF &1-&2 not applied.

**CPF35F4**  
Error occurred during cover letter processing.

**CPF35F6**  
MPTFI for library &1 deleted and created.

**CPF354A**  
Cannot specify \*SELECT for the CD-ROM path identifier.

**CPF354C**  
Cannot process PTF files on CD-ROM volume.

**CPF354D**  
Device &1 not allowed.

**CPF354E**  
No file selected.

**CPF354F**  
Required PTF file cannot be processed.

**CPF355B**  
Multiple releases for product &1 found on media.

**CPF355C**  
No PTFs found in path identifier &1.

**CPF3558**  
Cannot allocate &1 in &3 type \*&2.

**CPF3564**  
PTF &1-&2 damaged.

**CPF358A**  
Release not valid.

**CPF3586**  
List of PTFs not correct.

**CPF3587**  
PTFs not loaded.

**CPF3590**

PTF &1-&2 &3 not loaded.

**CPF3598**

PTF function already in process.

**CPF3606**

Product &1 &2 not installed.

**CPF361D**

Apply order of PTFs cannot be determined.

**CPF3612**

Library &1 not found.

**CPF3616**

No PTFs loaded.

**CPF3619**

PTFs for release &1 found on device.

**CPF3657**

PTFs not loaded because error occurred.

**CPF3693**

Service function ended because error occurred.

**CPF3924**

PTF not loaded.

**CPF3931**

Required programs not found. PTF incomplete.

**CPF3945**

Records of PTF activity for licensed program are deleted.

**CPF3992**

No PTFs exist on save/restore media for licensed program &1 &2.

**CPF6602**

PTF &1-&2 &3 not found.

**CPF8191**

Product definition &4 in &9 damaged.

**CPF8193**

Product load object &4 in &9 damaged.

---

## **LODQSTDB (Load Question-and-Answer Database) Command Description**

LODQSTDB Command syntax diagram

### **Purpose**

The Load Question-and-Answer Database (LODQSTDB) command allows the user to load a Question-and-Answer (Q & A) database from an alternative medium (such as tape) to the system. More information is available in the Basic System Operations topic in the Information Center.

### **Restrictions:**

1. This command is shipped with public \*EXCLUDE authority.

2. A user must have authority to the command and be a Q & A coordinator for any Q & A database referred to by the command.
3. This command is interactive only.

### Optional Parameters

#### QSTDB

Specify the Q & A database to be loaded to.

**\*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 to be used. If the Q & A database already exists on the system, the supplied subset of the Q & A database will be replaced. If the Q & A database does not exist on the system, it will be created in the specified library.

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

**QUSRSYS:** The library default for this command is QUSRSYS.

*library-name:* Specify the library where the Q & A database is to be loaded. The library must exist on the system.

#### Example for LODQSTDB

```
LODQSTDB
```

This command shows the Load Database to System display.

#### Error messages for LODQSTDB

---

## MRGMSGCLG (Merge Message Catalog) Command Description

MRGMSGCLG Command syntax diagram

### Purpose

The Merge Message Catalog (MRGMSGCLG) command merges message text from one or more source files (SRCFILE parameter) with message text in the specified message catalog (CLGFILE parameter). If the catalog specified does not already exist, it will be created using values specified for the CLGCCSID, DTAAUT, and OBJAUT parameters. If the catalog already exists, the CCSID, DTAAUT, and OBJAUT attributes of the existing message catalog will be used.

You can specify up to 300 message text source files. Message text source files are processed in the sequence specified. Each successive source file modifies the catalog. If a message number in the source file already exists in the message catalog, the new message text defined in the source file replaces the old message text in the message catalog file. If a message number in the source file does not already exist in the message catalog, the message information is added to the message catalog.

This command can also be issued using the following alternative command name:

- GENCAT

#### Example for MRGMSGCLG

```
MRGMSGCLG CLGFILE('/USDIR/USMSG.CAT') CLGCCSID(*SRCCSID)
          SRCFILE('/QSYS.LIB/MYLIB.LIB/MSGSRC.FILE/USMSG.MBR')
          DTAAUT(*R) TEXT('Message catalog for USA')
```

This command merges the message text from member USMSG of source physical file MSGSRC in library MYLIB in the QSYS file system with message catalog USMSG.CAT in directory USDIR. If the message

catalog does not already exist, it will be created with the CCSID of the source file and data authority of \*R. The text parameter describes this as a message catalog for the USA.

## Error messages for MRGMSGCLG

### \*ESCAPE Messages

#### CPF3BE3

Message catalog &1 not created or updated.

---

## MRGMSGF (Merge Message File) Command Description

MRGMSGF Command syntax diagram

### Purpose

The Merge Message File (MRGMSGF) command allows the user to merge messages from one message file with those in another message file. Another message file may be specified to hold the messages that are replaced during the merging process. None of the message files specified are deleted by this command (MRGMSGF).

Before the command is processed, messages can be in the from-message file (FROMMSGF), in the to-message file (TOMSGF), or in both files, but not in the replaced-message file (RPLMSGF). The three possibilities result in the following when the MRGMSGF command is processed:

- When the messages are only in the FROMMSGF, they are added to the TOMSGF
- When the messages are only in the TOMSGF, they remain in the TOMSGF
- When the messages are in both the FROMMSGF and the TOMSGF, the messages in the TOMSGF are first saved into the RPLMSGF (if a replace-message file is specified); then the messages in the TOMSGF are replaced by the messages in the FROMMSGF

**Restriction:** The user must have \*USE authority to the from-message file (FROMMSGF); use, add, and delete authorities to the to-message file (TOMSGF); and use and add authorities to the replace-message file (RPLMSGF).

### Examples for MRGMSGF

The examples below show how the merge files look before and after the MRGMSGF command is run.

#### Example 1: Merging Two Files

In the following example, two files are merged.

```
MRGMSGF FROMMSGF(A) TOMSGF(B)
```

**Table 1. Contents of the Files Before the Merge**

Message File A	Message File B
ABC1234 'text A4'	ABC1233 'text B3'
ABC1236 'text A6'	ABC1234 'text B4'
ABC1237 'text A7'	ABC1235 'text B5'
ABC1238 'text A8'	ABC1236 'text B6'

Below are the two message files after the MRGMSGF command is processed. Notice that messages ABC1234 and ABC1236 are in both files. When the merge occurs, the message text from file A (text A4 and A6 respectively) replaces the message text in file B (text B4 and B6 respectively).

**Table 2. Contents of the Files After the Merge**

**Message File A**

ABC1234 'text A4'  
ABC1236 'text A6'  
ABC1237 'text A7'  
ABC1238 'text A8'

**Message File B**

ABC1233 'text B3'  
ABC1234 'text A4'  
ABC1235 'text B5'  
ABC1236 'text A6'  
ABC1237 'text A7'  
ABC1238 'text A8'

**Example 2: Merging Two Files with Replace File Option**

In the example below, messages that are replaced in the to-file are saved to a separate file before being replaced.

```
MRGMSGF FROMMSGF(A) TOMSGF(B) RPLMSGF(C)
```

**Table 3. Contents of the Files Before the Merge**

**Message File A**

ABC1234 'text A4'  
ABC1236 'text A6'  
ABC1237 'text A7'  
ABC1238 'text A8'

**Message File B**

ABC1233 'text B3'  
ABC1234 'text B4'  
ABC1235 'text B5'  
ABC1236 'text B6'

**Message File C**

Below are the two message files after the MRGMSGF command is processed. Notice that messages ABC1234 and ABC1236 are in both files. When the merge occurs, the text from these two messages is first moved to file C (text B4 and B6 respectively). Then, message text from file A (text A4 and A6 respectively) replaces the message text in file B (text B4 and B6 respectively).

**Table 4. Contents of the Files After the Merge**

**Message File A**

ABC1234 'text A4'  
ABC1236 'text A6'  
ABC1237 'text A7'  
ABC1238 'text A8'

**Message File B**

ABC1233 'text B3'  
ABC1234 'text A4'  
ABC1235 'text B5'  
ABC1236 'text A6'  
ABC1237 'text A7'  
ABC1238 'text A8'

**Message File C**

ABC1234 'text B4'  
ABC1236 'text B6'

**Error messages for MRGMSGF**

**\*ESCAPE Messages**

**CPF2401**

Not authorized to library &1.

**CPF2407**

Message file &1 in &2 not found.

**CPF2411**

Not authorized to message file &1 in &2.

**CPF2452**

Replaced message file must contain no messages.

**CPF2461**

Message file &1 could not be extended.

**CPF2483**

Message file currently in use.

**CPF2510**

Message file &1 in &2 logically damaged.

**CPF2519**

Error occurred while processing message ID list.

**CPF2561**

Messages were not merged.

**CPF2562**

Cannot specify the same message file more than once.

**CPF9830**

Cannot assign library &1.

**CPF9838**

User profile storage limit exceeded.







Printed in U.S.A.